Presented at the 1997
International Conference on Mobile Planetary Robots
Santa Monica, California - January 1997

An Approach to Rough Terrain Autonomous Mobility

Alonzo Kelly and Anthony Stentz
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213
alonzo@ri.cmu.edu, http://www.frc.ri.cmu.edu/~alonzo
axs@ri.cmu.edu, http://www.frc.ri.cmu.edu/~axs

Abstract. Off-road autonomous navigation is one of the most difficult automation challenges from the point of view of constraints on mobility, speed of motion, lack of environmental structure, density of hazards, and typical lack of prior information. This paper describes an autonomous navigation software system for outdoor vehicles which includes perception, mapping, obstacle detection and avoidance, and goal seeking. It has been used on several vehicle testbeds including autonomous HMMWV's and planetary rover prototypes. To date, it has achieved speeds of 15 km/hr and excursions of 15 km.

We introduce algorithms for optimal processing and computational stabilization of range imagery for terrain mapping purposes. We formulate the problem of trajectory generation as one of predictive control searching trajectories in command space. We also formulate the problem of goal arbitration in local autonomous mobility as an optimal control problem. We emphasize the modeling of vehicles in state space form. The resulting high fidelity models stabilize coordinated control of a high speed vehicle for both obstacle avoidance and goal seeking purposes.

An intermediate predictive control layer is introduced between the typical high-level strategic or artificial intelligence layer and the typical low-level servo control layer. This layer incorporates some deliberation, and some environmental mapping as do deliberative AI planners, yet it also emphasizes the real-time aspects of the problem as do minimalist reactive architectures.

Keywords: mobile robots, autonomous vehicles, rough terrain mobility, terrain mapping, obstacle avoidance, goal-seeking, trajectory generation, requirements analysis

Table of Contents

1 Introduction

In earlier papers [32][33][34][35], the authors derived a set of requirements for autonomous off-road mobility that also suggest an approach to meeting those requirements. This paper is concerned with the design and implementation of a system that learns from the results of our earlier theoretical analysis. We have called this system RANGER - for Real-Time Autonomous Navigator with a Geometric Engine[1].

We emphasize the real-time nature of high speed autonomous mobility and, as a result, have been very concerned with such matters as efficiency, speed, throughput, and response time.

Our approach is based fundamentally on the state space representation of a multi-input / multi-output dynamical system and is a departure from precedent in the following ways:

- We use an active and adaptive approach to perception that minimizes the amount of perceptual data processed.
- We use a predictive control formulation of trajectory generation and search.
- We use an optimal control formulation of goal arbitration.

2 Local Autonomous Mobility

The navigator addresses what we will call the **local navigation problem** for autonomous vehicles. That is, the problem of deciding what to do based only on what can be seen at the moment within the field of view of the environmental sensors. The problem of global planning is outside our scope in this paper.

2.1 Introduction

Consider the task of path planning for an autonomous vehicle travelling cross country over rough terrain at high speeds.

2.1.1 Scope of Problem

We can scope the problem in terms of several parameters:

- **Overall Goal**. In general, the vehicle must achieve some useful goal. The goal may be to move from an initial position to some other distant position, to map an entire area, or to search an area for objects.
- **Degree of Optimization**. Standards for what constitutes achievement of the goal may vary from satisfaction of certain constraints (e.g. avoid collision with obstacles) to optimization of an arbitrary utility function (e.g. fuel consumption or distance travelled).
- **Difficulty of Terrain**. In realistic terrain, the vehicle is challenged by regions that would cause tipover, trapped wheels, or loss of traction. Some regions are not traversable at all and others may cause disastrous system failures such as falling into an abyss.
- **Depth of Prior Knowledge**. Both the goal to be achieved and the characteristics of the environment may be expressed and/or known to varying degrees of detail. The goal may be expressed as a point to achieve, a path to follow, an object to find, or something much more abstract. The environment may be completely known of partially mapped at various levels of detail and richness of expression.

2.1.2 Problem

Within these parameters, we can characterize the problem we have addressed in the following terms:

- The overall goal is to follow a predefined path that is assumed to be free of local planning minima.
- We attempt to follow this path as closely as possible while travelling as fast as possible and avoiding any obstacles that may appear.
- We attempt to operate on barren, rolling terrain that may contain ravines, cliffs, and regions that would tip the vehicle.
- We have no prior knowledge of the environment beyond the path we are to follow.

2.1.3 Previous Work

Groundbreaking work on this and related problems has been conducted at Hughes [10][9][13][36][45], JPL [41][55], CMU [16][20][22][23][24] and INRIA [28], among many others. Generally speaking, early systems were slower than later ones, and speed, excursion, and run time have improved with the general improvement in component technology and algorithms.

2.1.4 Solution

Our general approach to the problem has been to insert an architectural layer between strategic planning and actuator control which we will call **tactical control.** This layer, being faster than planning yet more intelligent than control will be able to understand vehicle maneuverability sufficiently well for robust path tracking and react fast enough for robust obstacle avoidance.

1. The name RANGER is also being used currently for an unrelated free-flying space vehicle under development by David Akin at the University of Maryland.

2.2 Preliminaries

2.2.1 Conventions

2.2.1.1 Lexical Conventions

The paper will introduce many new terms as a device to foster brevity and precision. New terms will be defined in their first appearance in the text. They will generally be highlighted **thus**, and will appear in a glossary at the end of the paper for easy reference.

2.2.1.2 Coordinate Conventions

The angular coordinates of a pixel will be expressed in terms of horizontal angle or **azimuth** ψ, and vertical angle or **elevation** θ. Three orthogonal axes are considered to be oriented along the vehicle body axes of symmetry. Generally, we will arbitrarily choose z up, y forward, and x to the right:

- x - **crossrange**, in the groundplane, normal to the direction of travel.
- y - **downrange**, in the groundplane, along the direction of travel.
- z - **vertical**, normal to the groundplane.

2.2.1.3 Notational Conventions

We will carefully distinguish range, R measured in 3D from a range sensor, and the projection of range Y onto the groundplane. Generally, both will be measured forward from the sensor unless otherwise noted.

2.2.1.4 Nomenclature

Certain vehicle dimensions that will be important are summarized in the following figure. One distinguished point on the vehicle body will be designated the vehicle control point. The position of this point and the orientation of the associated coordinate system is used to designate the pose of the vehicle.

The wheelbase is L, and the wheel radius is r. The height of the sensor above the groundplane is designated h and its offset rear of the vehicle nose is p. The height of the undercarriage above the groundplane is c. Range measured from the sensor is designated R.

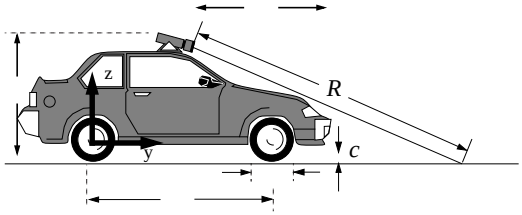


Figure 1: Important Vehicle Dimensions.

A complete list of symbols is provided at the end of the paper for easy reference.

2.2.2 Terminology

Any vehicle which attempts to navigate autonomously in the presence of unknown obstacles must exhibit performance that satisfies a basic set of requirements. At the highest level, if the system is to survive on its own, the vehicle control system must implement a policy of **guaranteed safety**.

This requirement to guarantee safety can be further broken down into four other requirements on performance and functionality expressed in terms of timing, speed, resolution, and accuracy. In order to survive on its own, an autonomous vehicle must implement the four policies of:

- **guaranteed response**: It must respond fast enough to avoid an obstacle once it is perceived.
- **guaranteed throughput**: It must update its model of the environment at a rate commensurate with its speed.
- **guaranteed detection**: It must incorporate high enough resolution sensors and computations to enable it to detect the smallest event or feature that can present a hazard.
- **guaranteed localization**: It must incorporate sufficiently high fidelity models of itself and the environment to enable it to make correct decisions and execute them sufficiently accurately.

2.2.3 Subproblems

Analysis shows [31] that traditional configuration space planning techniques applied to cross-country navigation suffer from problems of poor reliability and stability, and poor computational efficiency. Indeed, our experience has demonstrated regular collisions with obstacles that were seen and reacted to. There were two general reasons for this behavior:

- **computational inefficiency**: there was not enough time to decide what to do.
- **command following problem**: the specific trajectory used to avoid the obstacle could not be executed reliably or stably.

Yet another common problem is the well-known **local minimum problem**. It arises from the use of local rather than global optimization strategies. This problem is considered *outside the scope of our work here*, though one of the authors [53] addresses this problem in our target environment in other writings.

2.3 Standard Architectural Model

Consider the following hierarchical architectural model. This is a convenient view for organizing the description of the system.

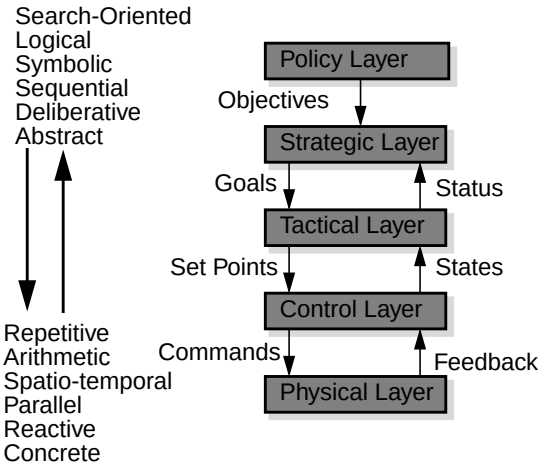


Figure 2: Standard Model. This type of hierarchical architecture is common. Higher layers tend to be more deliberative etc. whereas lower layers tend to be more reactive etc. Our control formulation of mobility fits in at the "tactical" level.

2.3.1 Spectrum of Characteristics

Higher levels of the hierarchy tend to be characterized by computation that is more symbolic, logical, search-oriented, sequential, deliberative and abstract than lower layers. Lower layers tend to be characterized by computation that is more spatial or temporal, arithmetic, repetitive, parallel, reactive, and concrete than higher layers. As a general rule, higher layers exhibit longer reaction times and longer cycle times.

The policy layer concerns itself with the generation and monitoring of mission level objectives such as "stay alive', "find the bomb", etc. Generally, the goals of this level are not subject to much compromise. Policy does not change often and is usually constant over the duration of a mission. Policy is often imparted permanently to a system by its human designers.

The strategic layer corresponds to the deliberative, logical, goal-generating component of autonomous systems. It concerns itself with the larger picture within the confines of policy; with avoiding local planning minima, with overall optimality, and with modeling and memory of the environment. Relative to lower layers, it is often obliged to consume more time in its deliberations of longer term strategic concerns.

By contrast, the control layer corresponds to the real-time command following component of autonomous systems. It concerns itself with the immediate low level issues of how much power should be applied to actuators and generally, with doing exactly what it is told to do to the best of its ability. Relative to higher layers, it is often obliged to consume more bandwidth as it reacts almost instantly to short term immediate concerns.

It is clear that both longer term strategic and shorter term reactive concerns will contend for computational resources. Plainly, there are limits to the degree to which any system can be both smart and fast. Faced with this reality, the design problem becomes one of making the best of available resources.

2.4 Tactical Control Layer

Much of our work resides in the layer we are calling **tactical control**. We identify this layer in order to more effectively connect the strategic and control layers, and to provide a place solve many of the problems mentioned earlier.

2.4.1 Strategic - Control Connection

A direct connection of the strategic layer (say, the global path planner) to the control layer (actuator control) becomes less feasible as speeds increase. Further, there are times when one or the other generates incorrect output and the other is either not intelligent enough or not fast enough to compensate.

There are certainly times when the goals specified by the strategic layer must be ignored because it is not aware of the immediate environment but the control layer is not intelligent enough to compensate. There are also times when the control layer is unable to follow its commands but the strategic layer is too slow to alter the command.

Our solution to this problem is to have a third layer more intelligent than control and faster than planning. This layer:

- Views the goals from the strategic layer as recommendations that it may be override when the situation demands it.
- Incorporates sufficient bandwidth to ensure vehicle survival at the coordinated actuator control level.
- Incorporates a sufficiently accurate model of vehicle dynamics that it understands and adapts to the inability of the controller to follow its commands.

This three layer architecture imparts a degree of autonomy to the layer below the strategic to allow it to implement basic survival a temporarily disregard strategic imperatives.

2.4.2 Intelligent Predictive Control

We characterize the navigator as an intelligent predictive controller because it closes the overall perceive-think-act loop for a robot vehicle based on intelligent assessment of both the surrounding environment and the abilities of the vehicle to maneuver.

The system shares many characteristics with strategic planning:

- It performs an amount of search and heuristics are employed to reduce that search.
- It models the environment and responds to it, so it merits the designation *intelligent*.
- It considers alternatives in an abstract space that is a transformation of reality - in this case, command space instead of the configuration space commonly used in strategic motion planning.
- It considers the consequences of its actions using time continuous precedence information in the form of a feedforward system dynamics model.
- It employs some memory of the state of the environment and of the vehicle.

Likewise, the system also shares characteristics with controllers.

- It models the vehicle with a multivariate differential equation.
- It is very concerned with response time and throughput management as is common of real-time systems.
- It is concerned with latencies and time tags and the precise timing of events.
- It concerns itself with command following - although it may temporarily override its commands.

2.5 Architecture

At the highest level, the system can be considered to consist of 5 modules as shown in the following data flow diagram:

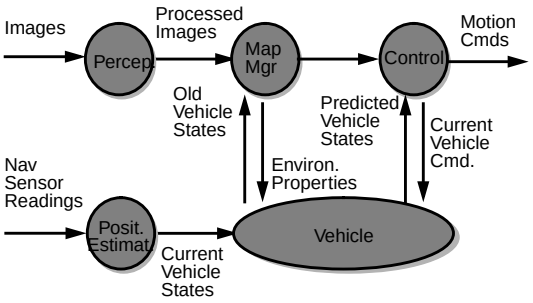


Figure 3: System Data Flow: Five components make up the tactical control layer.

2.5.1 Position Estimator

The Position Estimator is responsible for integrating diverse navigation sensor indications into a single consistent indication of vehicle state. Vehicle state information includes the positions of all actuators and some of their derivatives, and the 3D state of motion of the vehicle body. This module may be the built-in navigation Kalman filter or another system which generates the same output.

2.5.2 Map Manager

The Map Manager integrates discrete samples of terrain geometry or other properties into a consistent terrain map which can be presented to the vehicle controller as the environmental model. It maintains a current record of the terrain immediately in front of the vehicle which incorporates all images necessary, and which automatically scrolls as the vehicle moves.

2.5.3 Vehicle

The Vehicle object is both the control loop feedforward element and an abstract data structure which encapsulates the vehicle state. It incorporates FIFO queues which store a short time history of vehicle states and commands. Old state information is required by the map manager in order to register images in space. The current vehicle state and old commands are used in the feedforward dynamic simulation.

2.5.4 Controller

The Controller object is responsible for coordinated control of all actuators. This module includes regulators for sensor head control, an obstacle avoidance tactical controller and a path following strategic controller. It also incorporates an arbiter to resolve disagreements between the latter two controllers.

2.5.5 Perception

The Perception module is responsible for understanding or interpreting input images and putting them in a form suitable for the Map Manager to process. Examples of perceptual preprocessing include stereoscopy (stereo vision) which computes range from two or more images taken from disparate viewpoints, and terrain-typing which labels each pixel in an image as rock, road, shrubbery or tree. Stereo vision is the only perception that is currently supported, although laser range images can be fed directly to the map manager.

2.6 Results

The system has been tested on the vehicles shown in the following figure. We have used laser range data and stereo range data to build maps of the terrain over which the vehicle must travel. In the former case, excursions of 15 kilometers and instantaneous speeds of 15 km/hr have been achieved while tracking a coarsely specified path. Average speed was on the order of 7 km/hr.

Figure 4: Some Navigation Testbeds: A modified military HMMWV and a RATLER Planetary Rover prototype.

3 Perception

This section discusses the motivation behind, and implementation of the 3D perception algorithm for extracting relevant geometry from a range image sequence. We propose a relatively simple method of approaching the minimum required perceptual throughput in a terrain mapping system, and hence the fastest possible update of the environmental model. The technique proposed will be relevant to any application that models the environment with a terrain map or other 2-1/2 D representation.

3.1 Introduction

The surface of the surrounding terrain can be sensed by any number of means, but the two most commonly used ones in outdoor scenarios are laser rangefinders and stereo vision. We represent the surface of the surrounding terrain by a sampled, uniform density data structure often called a **terrain map** or **cartesian elevation map**.

3.1.1 Problem

When attempting to navigate over rough terrain, few assumptions about the shape of the terrain ahead can be made. It can be necessary to convert images into a full description of the geometry of the scene at relatively high rates. As a result, the speed of rough terrain navigation is typically limited by the throughput of the perception system. We will call this predicament the perceptual **throughput problem**. Perceptual throughput can be expressed in units of range or intensity pixels measured per second, or its equivalent.

We address here this typical performance limitation of autonomous outdoor vehicles. Analysis suggests [31] that much of the computational resources used to image and interpret the environment can be a waste of resources in mobility scenarios. This waste occurs for three principle reasons:

- The vertical field of view is often too wide from a throughput perspective. Obstacles and other hazards normally appear in the field of view long before they can be resolved, and long after they cannot be avoided.
- Sensor frame rate is often too fast. The sensor vertical field of view is normally aligned with the direction of travel so that image sequences normally contain much redundant information.
- Square pixels are not optimally shaped. The projection of image pixels on the groundplane is normally elongated in the wrong direction for robust obstacle detection and minimum throughput.

From the days of the Stanford Cart [44] to the Autonomous Land Vehicle [10], vehicle speed has been limited, at least in part, by limited perceptual throughput. We will show how to eliminate much of this inefficiency in order to generate perceptual throughput requirements that can be met easily.

3.1.2 Solution

One approach to reducing redundant information is the use of laser and video line scanners. These have seen use in specialized high-speed inspection applications for some time. In satellite applications, synthetic aperture radar has used vehicle motion to provide the scanning motion of the sensor along the direction of travel. The essential principle involved in these examples is to avoid scanning the sensor when either the motion of the vehicle or the motion of the environment already accomplish the scanning.

However, the use of line-scanned sensors is difficult on

rough terrain because abrupt attitude changes of the vehicle body cause holes in the coverage of the sensor. Software adaptation provides the best of both worlds because it gives the ideally focussed attention necessary for high speed and the wide field of view necessary for rough terrain.

In our earlier paper [34], we have showed that required perceptual throughput is polynomial in the reaction distance. This analysis suggests that certain approaches to mapping the environment in mobility scenarios can waste significant computational resources.

We have also showed that straightforward techniques promise to significantly increase the overall efficiency of terrain mapping perception algorithms. This improvement can be accomplished by exploiting constraints and several assumptions that are valid in most outdoor mobility scenarios.

The basic idea is to selectively process only the data that matters in range imagery. Known here as **adaptive perception**, the technique also has the beneficial side-effects of automatically adapting to changes in vehicle speed and attitude, and to the local slope of the imaged terrain. Through this technique we achieve near minimum perceptual throughput and hence, near maximum safe vehicle speeds.

A fundamental tenet of **active vision** [1][2] is to direct attention to the part of the scene that is relevant to the task at hand, rather than to interpret and model the scene. Our work provides a concrete example of at least one aspect of active vision. Although we do model the scene, we actively search for the data we need.

3.2 Preliminaries

We will use two primary techniques for reduction of the perceptual inefficiencies mentioned above:

- We will actively maintain a **focus of attention** and process perceptual data only in a **region of interest** that contains the most useful information.
- We will actively and intelligently subsample the data within that region of interest for adequate - but not unnecessarily high - resolving power.

These two strategies will be referred to collectively as **adaptive perception** - the organizing principle of our approach to terrain mapping for high speed mobility.

3.2.1 Terminology

We will call a region of space for which sensory data is required a **region of interest**, abbreviated ROI.

3.2.1.1 Regions of Interest

It also will be important to distinguish the coordinate system implied by the sensor image - called the **image plane** from a set of coordinates attached to the terrain - called the **ground plane**.

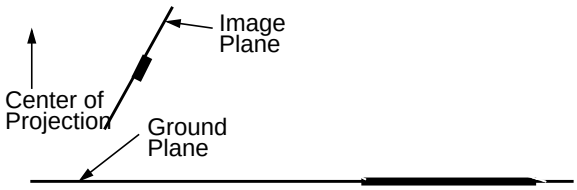


Figure 5: Regions of Interest. A region of interest in the ground plane forms a corresponding image in the image plane.

An ROI defined on the groundplane will be called a **ground plane ROI**. Such a region will have an image in the image plane which will be called an **image plane ROI**.

3.2.1.2 Differential Relationships

Let θ and ψ be the elevation and azimuth coordinates of an image pixel. Computing derivatives of the range image of flat terrain leads to the differential relationships between groundplane (x,y) resolution and image plane resolution (θ, ψ):

$$dx = Rd\psi \qquad dy = (R^2/h)d\theta$$

The completely correct transformations also depend on the local terrain gradients. These are unknown a priori because terrain geometry is the very thing the sensor is supposed to measure.

3.2.1.3 Response Distance

A quantity of central concern to us will be the distance that the vehicle requires to react to an external event such as the appearance of an obstacle in the sensor field of view. This distance will be called the **response distance** and its precise value will depend on:

- the speed of the vehicle when the event happens
- when the response is considered to be complete
- the maneuver chosen as the response

3.2.2 Subproblems

We have, at this point, nominated **adaptive perception** as a solution to the perceptual throughput problem. Unfortunately, this leads to a new set of problems, but we will be able to solve them with additional strategies and clearly identified assumptions.

3.2.2.1 Response - Resolution Tradeoff Problem

From the point of view of responding robustly to obstacles, it is best to detect obstacles early, or equivalently, at high range from the vehicle. However, from the point of view of sensor resolving power, it is best to detect obstacles as close as possible to the vehicle where data quality and spatial resolution tends to be highest. In other words, the farther away an obstacle is detected, the easier it is to avoid, but the harder it is to detect it robustly. When either resolution or range is

limited, we can detect an obstacle robustly or avoid it robustly, but not both. This is the **response-resolution tradeoff**.

We will manage this tradeoff by explicitly computing the minimum distance required for robust obstacle avoidance and looking for obstacles only beyond this distance. This technique will be called **adaptive lookahead**.

3.2.2.2 Selection Problem

The mapping from the groundplane ROI to the image plane ROI is both nonlinear (a projective transform) and a function of the unknown shape of the terrain. It seems, therefore, that it is not at all straightforward to efficiently find the image plane ROI. Consider, for example, the straightforward solution of converting coordinates of all pixels in the image and then comparing their positions to the groundplane ROI. After pixels that are not in the groundplane ROI are eliminated, one is left with the image plane ROI. While this would certainly work, it can be far too inefficient to be useful.

For terrain mapping, the largest computational cost of a range pixel is the conversion of its coordinates from the image plane to the ground plane. In attempting to select only the data of interest by converting the coordinates of all pixels, one has already done most of the perception task anyway. Any straightforward attempt to selectively process data in a region of interest apparently falters because *the problem of selection is as difficult as the problem of perception*.

We will use assumptions to decouple these problems. When the assumptions are combined with an appropriate choice of the groundplane ROI, we will be able to partially infer the shape of the image plane ROI and compute its position by very efficient image plane search. The algorithm for doing this will be called **adaptive sweep**.

3.2.2.3 Sampling Problem

The **sampling problem** is the nonuniform and anisotropic distribution of pixels on the groundplane which corresponds to a uniform and isotropic distribution of the corresponding pixels in the image plane. The Jacobian matrix which relates the two distributions depends on both the image projective transform and the local terrain slope at each point. The impact of this problem is that not only is the shape of the image plane ROI distorted and of unknown position but the local pixel density required to sample the groundplane uniformly is both unknown and different everywhere in the image plane ROI.

This variation in pixel density is shown below for flat terrain. Each ellipse represents the footprint of a pixel. It is the variation in density which we are illustrating, not the density itself, so the images were subsampled to avoid clutter.

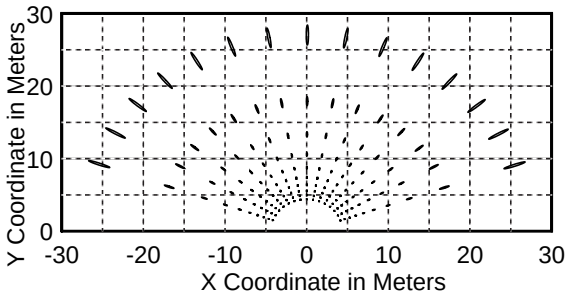


Figure 6: Sampling Problem. Equally spaced image pixels are not equally spaced on the groundplane - even for flat terrain. The situation is worse for rough terrain.

We will solve this problem to some degree by choosing the best compromise and, at other times, by actively computing the required image plane resolution from extrapolation. The algorithm for doing this will be called **adaptive scan**.

3.2.3 Assumptions

Certain assumptions will be key components of our approach - either because they must be made or because they can be made with little or no loss of generality.

3.2.3.1 Stationary World

One of our most fundamental assumptions will be that the environment is self stationary. That is, the environment will be supposed to consist of rigid bodies whose relative positions are fixed - at least while they are in the field of view of the environmental sensor. While the bodies comprising the environment are self stationary, our vehicle is in motion with respect to them. The value of this assumption is that it allows us to image a point in the environment only once and, because only the vehicle moves, its subsequent position relative to the vehicle at any later time can be inferred solely from the vehicle motion.

3.2.3.2 Small Incidence Angle

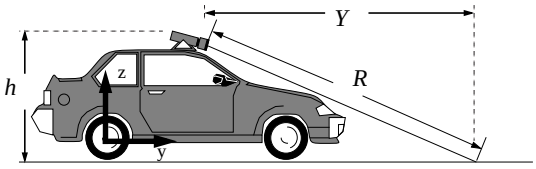
We will use the term **small incidence angle assumption** to refer to the situation where image pixels intersect a theoretical flat world at glancing angles. This is guaranteed to be the case if:

- the sensor is mounted on the vehicle roof, and
- pixels inside the response distance are ignored, and
- the vehicle speed is relatively high

because, under these conditions, the sensor height is small relative to the range of any pixel.

In the figure above, this assumption implies the validity of the following approximations:

We will call R the **range** and Y the **range projection**. It is easy to show that the relative error incurred in



$$Y^2 + h^2 = R^2$$

Figure 7: Imaging Geometry. The height of the sensor above the ground plane is normally small compared to the ranges measured.

$$\frac{h}{R} \ll 1 \qquad \text{and} \qquad Y \approx R$$

assuming that these two quantities are the same is the square of the ratio h/R. We will concentrate now on a specific class of region of interest - one that can be specified in terms of two range extremes. Let us define a **range window** or **range gate** as an interval given by R_{max} and R_{min} or, equivalently, by its corresponding range projection extremes Y_{max} and Y_{min}.

Suppose a pixel's range projection y is in a range projection gate:

$$Y_{min} < y < Y_{max}$$

Then, to first order, we have, under our assumption:

$$R_{min} < y < R_{max}$$

Which is to say that we can directly compare range pixel values (an image plane ROI) to a region of interest on the groundplane (a groundplane ROI) while incurring very little relative error. The small incidence angle assumption allows us to efficiently implement a test in the image plane of membership in a groundplane ROI. Under our assumption, it is not necessary to convert range pixel coordinates so it inexpensively decouples the problem of selection from that of perception. Only those pixels which satisfy the inexpensive image plane ROI membership test need have their coordinates converted for mapping purposes.

3.2.3.3 Near Monotone Range Assumption

At this point, we have an efficient test for membership in a groundplane ROI. However, it is still expensive to test every pixel in a range image against a range gate. A final important assumption is the assumption that the environment is 2-1/2 dimensional with respect to the direction of gravity. That is, at all points, a line aligned with gravity pierces the first reflecting surface of the environment at most once. This assumption justifies a terrain map representation and it also allows us to assume that range is a near monotonic function of image elevation angle. The worst case violation of this **monotone range assumption** is the reduction in range that occurs when a vertical surface is scanned as shown below.

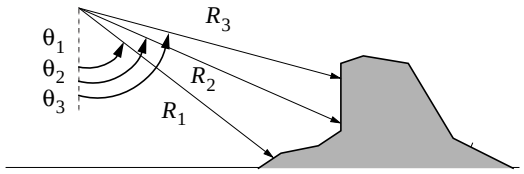


Figure 8: Nonmonotone Range. Range is a nonmonotone function of image elevation angle at a vertical or near-vertical surface.

The computational advantage of the assumption is that once the maximum range is found in an image, all pixels above it in the same column of the image can be safely assumed to be beyond that range. It will turn out later that this assumption will only be used in laser rangefinder implementations of adaptive perception. Stereo vision will not require it.

3.2.4 Design

Our adaptive perception algorithm confines the processing of range geometry in any cycle of computations to an image plane ROI with the following properties:

- It extends beyond the vehicle response distance.
- Its size is the distance moved since the last cycle.

The algorithm has three conceptual parts as outlined below.

3.2.4.1 Adaptive Lookahead

Adaptive lookahead means the process of adapting the position of the groundplane ROI to assure that there is sufficient time to react to hazards. There is some minimum range inside of which it is unnecessary to look because the vehicle is already committed to travel there. Also, there is some maximum range beyond which it is unnecessary to look because there will be time to look there later. In detail implementation, the algorithm can set the minimum range to the response distance, or alternately, set the maximum range to response distance plus the distance travelled per cycle.

3.2.4.2 Adaptive Sweep

Adaptive sweep is the process of adapting the width of the groundplane ROI to assure that there are no holes or excessive overlaps in the coverage of the sensor. The ROI width is set to the distance travelled since the last computational cycle. This determines both the maximum and minimum range projections in the groundplane and they are trivially converted to the image plane ROI based on assumptions mentioned earlier.

3.2.4.3 Adaptive Scan

Adaptive scan is the process of managing resolution within the image plane ROI in order to achieve uniform groundplane resolution. For the data of interest, it will be possible to compute an approximate mapping from groundplane resolution to image plane resolution and images will be subsampled by appropriate factors to achieve near uniform groundplane resolution.

3.2.5 Implications

Certain implications of using the adaptive perception algorithm are worth noting here.

3.2.5.1 Minimum Computational Cost Implies Highest Speeds

The minimum computational cost of this approach to perception has implications for the real-time performance of autonomous vehicles. The maximum useful range of a perception sensor is often limited by reasons of eye safety, computational cost, limited angular resolution etc. Given this limit, the highest safe vehicle speeds are normally achieved by minimizing reaction times. The only element of reaction time that can be changed easily is often the component due to the time required to process imagery or perform other computations. Therefore, to the degree that our approach minimizes the computational cost of perception, it also increases the vehicle speeds that can be achieved.

3.2.5.2 Adaptive Sweep Implies Image Stabilization

Our software adaptive approach to perception has the side effect of computationally pointing the sensor vertical field of view by responding to both changes in the vehicle attitude and changes in the shape of the imaged terrain. While the shape of the range window may be very irregular in image space, it always corresponds to a regular semi-annulus in the ground plane. If the vertical field of view is wide enough and the range sensor is fast enough in terms of range pixel rate, this software adaptation is superior to the technique of physically stabilizing the sensor because it responds instantaneously.

3.3 Adaptive Lookahead

The three techniques described in the previous section can be applied to any range image generated by an imaging laser or radar sensor or a stereo vision system. It is also possible to embed adaptive perception into a stereo vision algorithm - which will be the subject of a special section. For both classes of imagery, range imagery and stereo pairs, the adaptive lookahead algorithm is common.

A vehicle may attempt to turn to avoid obstacles and maintain its forward speed, it may elect to stop completely, or it may choose any other arbitrary trajectory. The choice of trajectory determines the details of computing the response distance. For our purposes, adaptive lookahead is implemented by computing the distance required to execute a 90° turn at the current speed. This gives the maximum range of the range window.

The groundplane ROI must be defined very precisely in terms of distances from some specific point on the vehicle at some specific time. The problem of finding the data in this region in an image taken previously involves several aspects of time delays and geometric offsets.

- The sensor is not mounted at the vehicle reference point, so the ROI is adjusted for this offset.
- The vehicle is not itself a point, so the ROI must be enlarged to provide data at the positions of the wheels forward and aft of the reference point.
- There may be significant delay associated with the acquisition of an image, so the ROI must be adjusted for the age of the image.
- The most recent vehicle state estimate is itself somewhat old and computation takes finite time. The ROI may need to be adjusted for these effects depending on the instant with respect to which the ROI is defined.

3.4 Adaptive Sweep/Scan-Range Imagery

If one starts with a dense range image, the algorithm consists of the mapping of the range window into image space and the extraction of the data.

3.4.1 Adaptive Sweep

Terrain roughness and nonzero vehicle roll mean that the position of the range window in the image is different for each column so the range window is processed on a per column basis. In order to robustly find the range window, each column is processed in the bottom-to-top direction.

A conceptual C code fragment is as follows. The image itself is of dimensions rows by cols. A constant rectangular subwindow of the image is searched which is delimited by the image plane coordinates start_row, start_col, end_row, and end_col. This region is known to always contain the ROI.

The **monotone range assumption** appears as the break statement after the first conditional of the inner loop. The start_col and end_col variables implement a fixed azimuth and elevation angle window within which the range window always lies on typical terrain.

3.4.2 Adaptive Scan

The variables row_skip and col_skip have values corresponding to the constant image subsampling factors that give the most acceptable groundplane resolution. In the case of range images, adaptive scan is implemented by a literal subsampling of the image. Also,

```
j = start_col;
while (j <= end_col+col_skip)
   {
   i = end_row;
   while (i >= start_row-row_skip)
      {
      R = range(i,j);
      if (R > Rmax )
         break;
      else if( R < Rmin )
         {i -= row_skip; continue;}
      else process_pixel_into_map();
         i -= row_skip;
      }
   j += col_skip;
   }
```

Figure 9: Adaptive Sweep Algorithm. The range window is processed on a per column basis in order to robustly extract the data of interest.

this subsampling applies to both the data in the ROI and the data below the ROI that is not processed. That is, adapting the resolution can benefit the speed of handling both the processed and the unprocessed data.

Because the differential transformation from the image plane to the groundplane is unknown, a perfectly robust, optimal subsampling solution is not available. However, a spectrum of approaches to resolution management are available based on the frequency of update of the row_skip and col_skip variables and how they vary with range for an assumed flat world. They can be computed based on:

- the highest projected value of the ROI maximum range, Rmax, based on the known speed limits of the vehicle.
- the value of ROI maximum range, Rmax, for the current computational cycle.
- the instantaneous value of range, R, at the current pixel.

These options have been listed in order of increasing speed and decreasing robustness.

In the least adaptive form of adaptive scan, the number of pixels skipped in the horizontal and vertical directions can be set based on the average or worst case expected value of the maximum range.

$$d\psi = \left(\frac{1}{R_{max}}\right)dx \qquad d\theta = \left(\frac{h}{R_{max}^{\ 2}}\right)dy$$

In the next most adaptive form, the image plane resolutions are recomputed for each image based on the current ROI maximum range. In the most adaptive form, image plane resolutions can be recomputed based on the instantaneous range image values. However, it can be awkward to vary the azimuth resolution as a function of range if one chooses to process the image by columns.

The ratio of maximum to minimum range is normally small, so the variation in $d\psi$ (row_skip) is also small. Under this assumption, a good compromise is to use the worst case azimuth resolution and the instantaneously computed elevation resolution.

$$d\psi = \left(\frac{1}{R_{max}}\right)dx \qquad d\theta = \left(\frac{h}{R^2}\right)dy$$

Although the flat world assumption may seem inappropriate on rough terrain, the use of it in adaptive scan works well in practice.

3.5 Adaptive Sweep/Scan-Stereo Imagery

The principles of the earlier section could be applied directly to the output of a stereo vision system. Yet, because stereo also consumes computational resources, it seems worthwhile to investigate whether similar techniques can be employed inside of the stereo algorithm itself in order to avoid computing range pixels that subsequently would be eliminated anyway.

Traditionally, the stereo problem is cast as one of determining the range for every pixel in the image. Traditional stereo finds the range for each possible angular pixel position. Conversely, our adaptive approach to stereo finds the angular positions in the image plane of each possible range value. It determines those pixels whose range value falls within a small range window, and it does so without computing the ranges of pixels which are not of interest. This principle is sometimes called **range gating** in laser rangefinders which employ it.

The motivation for the approach in the case of stereo is the observation that the region of terrain which is beyond the vehicle response distance usually corresponds to a very narrow range in stereo disparity space. The nonlinear relationship between range and disparity also implies that range resolution is relatively poor at high ranges, so the computation of the range of low range pixels can be wasteful. However, as before, the problem of selection, of determining membership in a range gate without computing the range, seems difficult.

3.5.1 Embedded Adaptive Sweep in Stereo Vision

For stereo ranging systems, the basic principle of the range window can be converted to a **disparity window**[1] for a stereo system because the range and disparity are related by the stereo baseline.

3.5.1.1 Disparity Window

The basic stereo configuration for perfectly aligned

1. There is a slight difference in the geometry of a stereo range image (perspective) compared to a rangefinder image (spherical polar). Therefore, a disparity window corresponds to a window on the y coordinate and not the true polar range. In most circumstances, this distinction can be safely ignored.

cameras is given below. It is useful to remove the

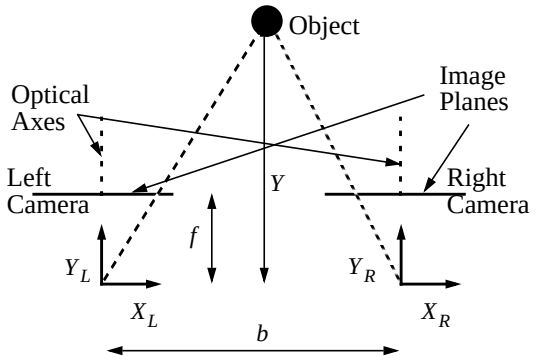


Figure 10: Stereo Triangulation. The relationship between disparity d, range Y, baseline b, and focal length f is derived from similiar triangles.

dependence of disparity on the focal length by expressing disparity as an angle. Define the **normalized disparity** thus:

$$\delta = \frac{d}{f} = \frac{b}{Y}$$

Then, for a range window between 25 meters and 30 meters, and a stereo baseline of 1 meter, the angular width of the corresponding disparity window is:

$$\Delta\delta = \frac{1}{25} - \frac{1}{30} = 0.0067 = 0.38°$$

Thus, the range of disparities which corresponds to a typical range window is roughly 1% of a typical camera field of view (40°). In other words, the image coordinates of corresponding points in both images are very close to each other if the range of the point is beyond the response distance.

3.5.1.2 Local Minimum Problem

In traditional area-based stereo, correlations (or any of a number of other measures of similarity of two image subwindows) are computed for a wide range of disparities. Then the algorithm searches along the curve generated for each pixel for the disparity, d^*, corresponding to the global correlation maximum. The case for normalized image crosscorrelation is illustrated below.

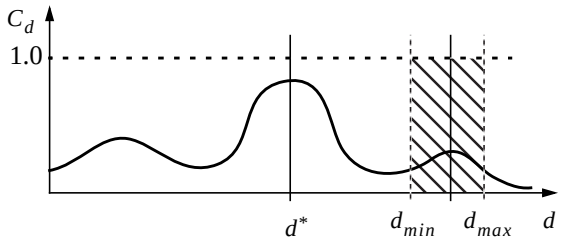


Figure 11: Disparity Search. The global maximum correlation is the best match. Limiting the search can lead to the wrong answer.

If, however, the search were limited to the disparity window whose boundaries are d_{max} and d_{min} in the above figure, the point of maximum correlation that would be found would only be a local minimum. No information other than the absolute value of the measure of similarity would indicate this. If a range image were generated based on the results of this limited disparity search, the image would contain:

- correct ranges for pixels whose true range happened to fall within the range window searched.
- incorrect ranges for pixels like the one illustrated above which defeated our best attempts to identify them at this stage of processing.

Nevertheless, the environment is often smooth, and this smoothness leads to the property that correct ranges tend to form large smooth regions whereas incorrect ones do not as illustrated below.

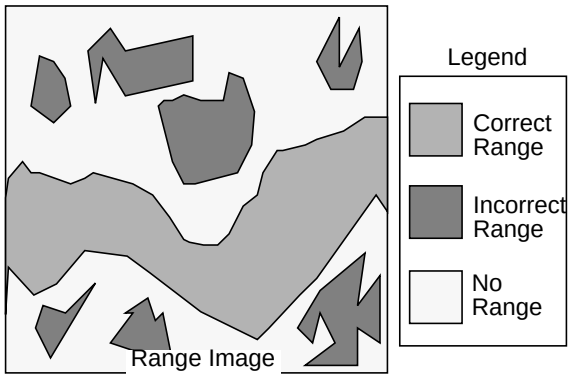


Figure 12: Spurious Disparities. Correctly ranged pixels tend to form large connected smooth regions. Incorrect ones do not.

It is well known that spurious matches occur fundamentally because regions which do not correspond physically actually look more or less the same. Several solutions to this repetitive texture problem help the situation somewhat but the simple technique of computing connected components and removing small regions [43] works effectively and is computationally free because a disparity image cleanup pass is required even when a wide disparity range is searched.

3.5.2 Embedded Adaptive Scan in Stereo Vision

In the case of stereo vision, the situation for adaptively changing resolution is more complex because range resolution and angular resolution are coupled. That is, once angular resolution is fixed, range resolution is also fixed, yet each has independent constraints imposed on it by the application. It is not possible, for instance, to aggressively reduce horizontal image resolution (as would be done with a range image) at the input to stereo because range resolution will also be dramatically and unacceptably degraded.

The least that can be done, however, is to compute the degree to which the output range image would be subsampled and then the latter stages of stereo (the stages past the correlation computation) can simply ignore the unwanted pixels. Before correlation, those unwanted pixels may be needed to participate in computing the correlations.

3.6 Results

The following two sections present performance results for adaptive perception based on laser range images and stereo vision. For these results, the vehicle speed is 3 meters/second and the resolution of the generated terrain map is 0.75 meters in both horizontal directions. An oversampling factor of 2 is also incorporated into adaptive scan as a safety margin to protect against terrain undersampling.

While adaptive perception resamples a range image for optimum coverage of the terrain, the specific attributes of the range sensor and cameras used for the following results are given in the table below:

Table 1: Sensor Parameters

Attribute	ERIM rangefinder	CCD camera
Image Rows	64	640
Image Cols	256	486
Hor. Field of View	80°	20°
Vert. Field of View	30°	20°
Hor. Angular Res	0.3125°	0.0412°
Vert. Angular Res	0.4688°	0.0312°
Frame Rate	2 Hz	30 Hz

3.6.1 Range Image Adaptive Perception

In a typical image, the pixels that are actually processed by the adaptive perception algorithm form a horizontal band that is jagged-edged and of varying width. The width of the band decreases if the vehicle speed increases because adaptive lookahead will move the window up in the image where a smaller width projects onto the same groundplane distance.

The following figure gives a sequence of range images for a run of our navigation system simulator[1] on very rough terrain using a simulated rangefinder where the pixels that were actually processed fall between the thin black lines. On average, only 75 range pixels out of the available 10,000 (or 2%) were processed per image. In terms of areas imaged per second, the system throughput is increased by a factor of 100 times, or two orders of magnitude.

There are five range images arranged vertically on the left. These are rendered as intensity images where darker greys indicate increasing distance from the sensor. The terrain map constructed by the perception system is rendered on the right. The top figure shows the map as an image where lighter greys indicate higher elevations. In the center of the map is the vehicle at the position where the 5th image was captured. The lower right figure is the same terrain map rendered as a wireframe surface from the vantage point of the initial position.

There are three hills in the scene whose range shadows are clearly visible in the terrain map. In the first image, the vehicle is accelerating but still travelling relatively slowly. The range window is relatively wide and positioned near the bottom of the image. The first hill is in the range window. In the second image, the second hill is in the range window and the first hill has already been processed. In the third image, the third hill is now in the range window. In the fourth image, the vehicle is driving past the first hill and is rolled to the right because of it. This rolls the image to the left and the algorithm compensates appropriately. In the fifth image, the range window has moved past the third hill to the flats beyond and a fourth hill is barely visible in the distance.

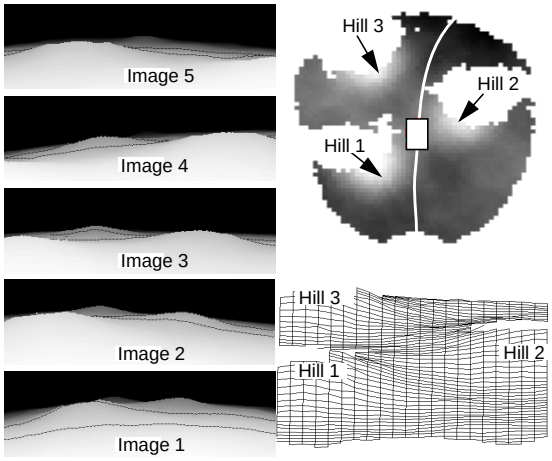


Figure 13: Adaptive Rangefinder Perception. The processing of five range images is illustrated as the vehicle drives through an obstacle course of three hills.

1. The system performs identically on real images but simulated ones were used here in order to illustrate several points in limited space.

Actual perception performance is given in the tables below for a series of images of flat terrain. In the table, the nonadaptive value corresponds to the result obtained by processing all pixels in the ERIM range image. The adaptive value is the value obtained by our range image algorithm:

Table 2: Rangefinder Adaptive Perception Performance (SPARC 20)

Attribute	Nonadaptive	Adaptive
Pixels Processed Per Image	16384	75
Run Time	0.352 secs	0.022 secs

The results do not scale linearly with pixels processed because the adaptive result includes a constant setup time. Nonetheless, the adaptive result is 16 times faster than the nonadaptive result and if the ERIM sensor had higher angular resolution, the improvement would be proportionally better. The system uses barely adequate spatial resolution and eliminates redundant measurements and hence achieves minimum throughput.

3.6.2 Stereo Vision

The following figure illustrates the operation of embedded adaptive stereo on two horizontal baseline input images. These are images of a barren ravine road near CMU taken from inside the ravine. The initial input images appear at the left. To the right of these are the nonadaptively processed disparity and range images. To the extreme right are the adaptively processed disparity and range images. The disparity images are shown to demonstrate the spurious matches which are caused by incorrectly chosen extrema in the correlation versus disparity curves.

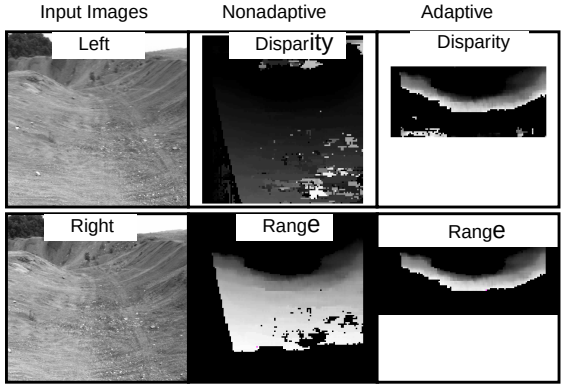


Figure 14: Adaptive Horizontal Baseline Stereo. The incorrect disparities due to incorrect matches are cleaned up with an efficient filter.

A breakdown of this run is shown in the table below:

Table 3: Stereo Adaptive Perception Performance (SPARC 20)

Attribute	Nonadaptive	Adaptive
Output Rows	120	48
Output Cols	128	128
Disparities	60	10
Preprocessing	102 msecs.	41 msecs.
Correlation	683 msecs.	69 msecs.
Postprocessing	754 msecs.	74 msecs.
Total Runtime	1539 msecs.	203 msecs.

4 Terrain Mapping

This section discusses the motivation behind, and implementation of our highly efficient approach to terrain mapping. We discuss methods for elimination of the need to copy and/or interpolate the data structure to incorporate incoming new data, methods to compensate for sensor motion, and methods for the representation of terrain shape uncertainty.

4.1 Introduction

Terrain mapping is the process by which surface descriptions, obtained from different vantage points, are accumulated into a consistent environmental model [22]. In order to provide the rest of the system with a single, coherent, uniform density data structure we transform images into a regularly-spaced cartesian grid called a **Cartesian Elevation Map** (CEM) or simply a **map**.

4.1.1 Problem

In our early attempts to map terrain for a fast moving outdoor vehicle, we encountered severe computational inefficiency problems for several reasons:

- The treatment of the motion of the vehicle through the environmental model necessitated a physical shift of data that was very expensive.
- Interpolation of the values of unknown cells from their neighbours was very expensive.
- Massive distortions of reality due to sensor motion were introduced as the vehicle speed increased.

4.1.2 Solution

We have developed methods to manage these problems that include:

- A special terrain map data structure and access routines.
- Real-time methods for processing sensor data.

4.2 Preliminaries

4.2.1 Terminology

In the map, each cell encodes z_{ij} where the z coordinate is unique for any pair i,j and is referenced to some fixed coordinate system called the navigation coordinate system with respect to which the vehicle moves. Individual elevation buckets in a terrain map are called **cells** to distinguish them from range image **pixels.**

4.2.2 Subproblems

Once we have implemented a wrapping terrain map data structure, we will face a new problem in distinguishing data from two different regions of space that happen to fall into the same cell. We will be able to manage this problem through the introduction of a new data field - the “age” of a cell.

4.2.3 Assumptions

Under some circumstances, natural outdoor terrain is well approximated by a surface expressed as z = f(x,y) where the z axis is aligned with the local gravity vector. An important exception to this assumption is trees and other large vegetation. We will assume that either we operate in barren terrain or that we can safely fill in the space beneath branches in our models. Thus, the use of a terrain map normally means that the **2-1/2 D world assumption** is being adopted.

4.2.4 Design

Our implementation includes the following elements:

- A 2D ring-buffer implementation of a terrain map that accommodates vehicle motion through modulo arithmetic indexing.
- Methods for processing perceptual data that never require copying, traversal, or interpolation of the terrain map.
- Straightforward methods to compensate incoming geometry for camera motion.

4.2.5 Implications

Before these methods were first adopted, over half of our processor time was consumed in simply managing the terrain map. That is, map management was more expensive than perception and planning combined. After they were adopted, the cost of terrain map management became so small that it was insignificant.

4.3 Wrappable Map

Using 30 meters of lookahead in planning, 1/6 meter resolution, and 20 bytes of memory per map cell, over 1/2 megabyte of memory is required to store a typical map. If this map is stored as a physically coherent block of memory, it must be physically shifted and copied after the acquisition of each image in order to account for the relative motion between the vehicle and the terrain.

4.3.1 2D FIFO Queue

Our solution to this problem is a classical one from computer science - the FIFO queue. A simple array accessed with modulo arithmetic suffices to *logically scroll* the map as the vehicle moves by *physically wrapping around* in memory. As in all FIFOs, the queue size must be chosen to exceed the worst case amount of memory required.

Let the rows and columns of the terrain map be aligned with the axes (x, y) of the navigation frame and be divided into cells of resolution dx by dy. Let the map width and height be W and H respectively. The indices into the array are determined by modulo arithmetic as follows:

$$map\left[\left\lfloor \frac{rem\left(\frac{x}{W}\right)}{dx} \right\rfloor\right]\left[\left\lfloor \frac{rem\left(\frac{y}{H}\right)}{dy} \right\rfloor\right] = z(x, y)$$

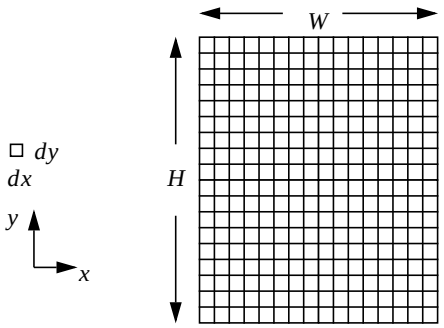


Figure 15: Wrappable Map Indexing. Using modular arithmetic, all of 2D space maps, with wraparound into a finite map.

The operator $\lfloor x \rfloor$ is the least integer function and $rem(x/y)$ is the floating point remainder function.

The operation of the technique when applied to three successive images is indicated below or both a physically scrolling and a wrappable map data structure.

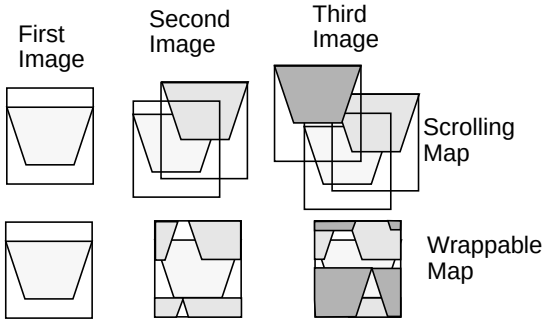


Figure 16: Wrappable Terrain Map. Data remains in the map until it is overwritten by incoming data from somewhere else that happens to fall in the same place in memory.

4.3.2 Cell Tags

This approach creates new problems. The mapping from world coordinates to map indices is multiply defined and therefore the inverse mapping is not a function. In mathematical terms, the coordinate transform is not **onto**.

An infinity of points in global coordinates correspond to a single cell in the map, so remnants of images of arbitrary age may remain in the map indefinitely. Suppose the elevation at the point (15, 25) is needed and the map is 10 by 10. Then the point (5, 15) may also be in the map. A query for the elevation at (15, 25) may get the elevation at (5, 15) instead.

We manage this problem in a very simple way. Although all data remains in the map until it is overwritten, each entry is tagged with the distance that the vehicle had travelled since the start of the mission at the time the pixel was measured. The interface routines then perform two important hidden functions:

- If the tag of the last update is too old, the interface routines report the cell as empty. This makes it impossible for old data to poke through the holes in new data.
- When the tag of new incoming data is significantly different from the one in the cell, it indicates wrap-around, so the statistical accumulators in the cell are first cleared. This ensures that two physically distinct regions of space are not confused and merged together.

4.4 Sensor Motion Compensation

By the time an image is received by the perception system, the vehicle may have moved a considerable distance since the image was acquired. So, the processing of the geometry in the image must account for the exact position of the vehicle *when the image was taken*.

Further, some sensors such as scanning laser rangefinders may require significant time to scan the laser beam over the environment. In the worst case, there is a distinct vehicle pose associated with each pixel in a ladar image. If this motion is not accounted for, the terrain maps computed from images will be grossly in error.

4.4.1 Smear and Offset

The worst case is a high angular velocity turn. If rangefinder scanning takes about 0.5 secs and the vehicle is travelling at 6 mph and turning sharply, its angular velocity can be as high as 1 rad/sec, so an obstacle can be smeared by 30° in a rangefinder image at high speed. Similarly, if the input latency is 0.5 secs and it is not accounted for, objects will also be shifted by 30° in a rangefinder image at high speed. Of course, the range to an object will also be overestimated by the distance travelled in 1 second.

4.4.2 Pose History and Lookup

We remove this distortion of range images is removed by maintaining a history of vehicle poses sampled at regular intervals for the last few minutes of execution. When a pixel is processed, we search the vehicle pose FIFO for the precise vehicle position at which each range pixel was measured.

4.5 Interpolation

The terrain map is not interpolated at all because interpolation requires a complete traversal which is too expensive to perform. Instead, the responsibility for interpolation is left with the users of the map.

4.5.1 Impact of Vehicle Maneuverability

Spatial interpolation of the entire map is wasteful because vehicle maneuverability constraints may prevent many places from being reachable. Hence, the data in such regions is not necessary at all and interpolating there is a waste of resources.

4.5.2 Impact of Occlusion

Note that occlusion is inevitable in rough terrain, so spatial interpolation can never succeed fully without unjustified and harmful smoothness assumptions.

4.5.3 Temporal State Interpolation

We will see later that the path planner *interpolates vehicle state in time* instead of interpolating the map in space. Further, the assessment of hazards is based on a time signal which may or may not be known at a particular point in time. The system is robust by design to unknown signal values and, as a by-product of its processing, computes an assessment of how much geometry is actually unknown and reacts accordingly.

4.6 Errors and Uncertainty

Practical solutions require methods to deal with both systematic and random error sources that corrupt the incoming data and its eventual representation.

4.6.1 Image Registration

A simple image registration algorithm is used in situations where edge artifacts are introduced by various forms of position and range sensor errors. The basic mechanism is to compute and remove the average elevation deviation between the overlapping regions of consecutive images.

Currently, only the elevation, z coordinate is matched and this seems to work best in practice. When the z deviation of two consecutive images is computed, it is applied to all incoming geometry samples in order to remove the mismatch error.

4.6.2 Elevation Uncertainty

After the mean mismatch error is removed, there are still random errors in the elevation data. In order to represent the variation in geometry in a single map cell and to improve signal to noise ratios, a **scatter matrix** is computed [28] incrementally as each new range pixel is merged into the map. The scatter matrix is defined as:

$$S = \begin{bmatrix} \sum xx & \sum xy & \sum xz \\ \sum yx & \sum yy & \sum yz \\ \sum zx & \sum zy & \sum zz \end{bmatrix}$$

The advantage of this incremental approach is that the mean and standard deviation of the evolving 3D distribution is available at any point in time from some simple formulas. Specifically, the deviation in z is useful for computing the uncertainty in the hazard estimates generated by the path planner.

4.7 Results

RANGER has been integrated with a stereo vision system at the Jet Propulsion Laboratory [41] on a HMMWV. The following figure shows a short autonomous excursion along a dirt road bounded by trees and bushes on the right and a ravine on the left. The sequence of images to the left are the stereo range images. To the right are intensity images of the scene corresponding to the range images.

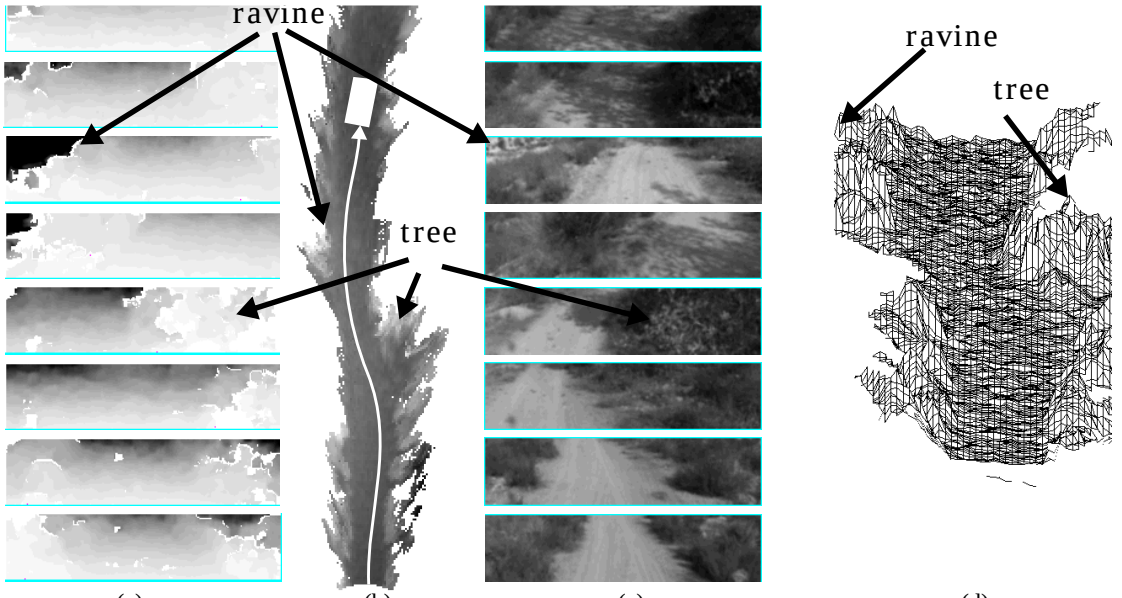


Figure 17: A short cross country excursion. (a) shows a sequence of range images from a stereo vision system mounted on a HMMWV vehicle. (c) shows a sequence of intensity images from one of the cameras. (b) and (d) show an overhead view of an elevation map that was generated.

The images are positioned in correspondence with their associated position in the terrain map. The terrain map, drawn in the center, is rendered with intensity proportional to elevation. The path followed is drawn leading to the position of the vehicle near the end of the run. The run terminates at the end of the road. Two distinct obstacle avoidance maneuvers occur. The first is a left turn to avoid a large tree and the second is a recovery right turn to prevent falling into the ravine.

5 Path Planning

This section discusses the motivation behind, and the implementation of our predictive control approach to trajectory representation, generation and search. We will call the collection of these three capabilities **path planning** in our context.

Our approach will be a departure from precedent that formulates the classical planning problem of deciding where to go largely in terms of *predictive control*. This approach will have advantages whenever speed is high enough for dynamics to matter or when nonholonomic motion constraints are operative. Our approach is similar to the approach to cluttered environment planning adopted in [15] and it echoes earlier work presenting a duality between feedforward control and deliberative planning [46].

5.1 Introduction

The local intelligent mobility problem can be characterized in terms of a search of the immediately visible environment, scanning for hazards, and seeking a goal while simultaneously avoiding any hazards that appear. Regardless of many other design variables, all or part of the local environment is typically searched.

5.1.1 Problem

Our initial attempts to search trajectories were founded on classical C-space techniques [40]. These attempts were very slow, brittle, and inelegant, but they were educational. In our early work, we encountered the following major problems:

- **computational inefficiency**: There was not enough time to decide what to do. Conversely, the inefficiency of computations limited vehicle speeds that could be safely achieved.
- **command following problem**: Specific trajectories used to avoid obstacles often could not be executed reliably or stably.

Our computational inefficiency problem was caused by a treatment of vehicle trajectories that was expensive and often wasteful. Our command following problem arose from issuing commands to the vehicle that were either wrong or unrealistic.

Further, consideration of these unrealistic trajectories in search tended to waste computational resources, thereby increasing reaction time and aggravating the first problem of computational inefficiency.

After conducting a study of some related real-time issues [32], we concluded that classical C-space planning techniques were ineffective in our domain. A new approach was necessary.

5.1.2 Solution

Motion planning is a problem involving search. Recall that heuristic search efficiency can be improved by appropriate ordering of constraints because some have more power to limit search than others. Our **predictive control** formulation amounts to a constraint ordering heuristic that improves the efficiency of search. The elements of our approach are:

- We represent trajectories *implicitly* in terms of the commands that are issued to the vehicle actuators, and ...
- The corresponding spatial trajectory is computed from a highly accurate state space vehicle model that guarantees mechanical feasibility by construction.

Through this technique we completely bypass many of the difficulties of trajectory generation and search for nonholonomic vehicles with real actuator response characteristics.

5.2 Preliminaries

Before proceeding to describe our technique, a few necessary terms will be defined.

5.2.1 Terminology

We will use a single point on the vehicle called the **reference point** to describe its motion. A spatial description of the continuous sequence of positions achieved or considered will be called a **path**, and when the time dimension is added, a **trajectory**.

5.2.1.1 Mechanical Feasibility of a Trajectory

The **kinematics constraint** is a term used to express the fact that steering mechanisms may be unable to achieve arbitrarily small curvatures. Any path which respects these mechanical limitations of the steering system is said to be **kinematically feasible**.

The **dynamics constraint** is a catchall term used to express the fact that system behavior is governed by differential equations. Any trajectory which satisfies this set of constraints is said to be **dynamically feasible**.

A trajectory is **mechanically feasible** if it is both kinematically and dynamically feasible. Such a trajectory describes a physically achievable motion.

A trajectory which is safe for the vehicle to execute (i.e free from significant hazards) is called **admissible**.

5.2.1.2 Representations and Spaces

It will be necessary to distinguish several alternate forms of trajectory representation. In general, the system has available to it at any time a space of possible commands that it can send to the vehicle controller for execution. This space of commands will be called the **command space** and a point in this space is a **command vector**.

Commands may or may not map directly onto the vehicle actuators, but when they are expressed directly in terms of actuated variables, they span the **actuation space**. In our case, vehicle commands map more or less directly onto the speed and steering actuators.

When these commands are applied to the vehicle actuators, the vehicle kinematics, dynamics, and the mechanics of terrain following cause it to traverse a unique trajectory. We can represent this trajectory in terms of the time evolution of a vector quantity called the vehicle **state vector,** which spans an abstract **state space**.

Nominally, the state vector includes the position and orientation of the vehicle body, and the positions of any articulations. Depending on the order of the system dynamics, it will also include time derivatives.

If time dependence is removed from the description by

representing only the geometry of the motion and time derivatives are eliminated from the description, the resulting description turns out to be a **configuration space** (C-space).

5.2.1.3 Command Versus Response

A distinction which is independent from representation is the distinction between *command* and *response*. The first is a specification of requested motion whereas the second is the actual response to that request. The degree to which these two agree is one measure of the fidelity of control that has been achieved.

5.2.1.4 Duality of States and Events

It will be useful to distinguish two approaches to dynamic system modeling that are analogous to the duality between actions and states that has been well discussed in the AI literature [25].

In a *state based representation*, system motion is viewed as a series of states that are altered by events. In an *event based representation*, the state of the system is derived implicitly from its initial state and all events that have occurred up to a particular point.

While the choice of one representation over another does not affect the capability for expression, it does affect the relative ease with which certain properties are represented and reasoned about.

5.2.1.5 C-Space Methods

Robot planning has commonly used an abstraction known as **configuration space** (C-space) [40] - a space spanned by any set of parameters that uniquely describe the configuration of the robot.

Relatively speaking, the determination of trajectory admissibility (safety) is fairly trivial in C space but significant work is required to ensure feasibility.

C-space methods, like state based methods, require an **inverse model** that computes the command trajectory that corresponds to the chosen C-space trajectory.

Such a model cannot be easily inverted and it may not be invertible at all for an arbitrary C-space curve. While C-space planning is a powerful paradigm, it is not an effective technique in problems where the model is difficult to invert. Such situations include cases where the system:

- requires a dynamic model[1], or
- is nonlinear, or
- is underdetermined (nonholonomic).

1. We will consider any situation where a derivative is required in a state vector to compute accurate trajectories to be one where a "dynamic" model is required. The need for one increases as speed increases.

5.2.1.6 Command Space Methods

The command space described earlier represents the space of alternative commands that the vehicle can receive. Such commands can always be executed in the sense that a legitimate, unique, computable response exists for all commands - although the response may not follow the command closely.

In this case, the determination of trajectory feasibility is fairly trivial whereas it requires some work to ensure that it is admissible.

Command space methods, like event based methods, require a **forward model** that computes the C-space trajectory that corresponds to the chosen command trajectory. Both forms of model are indicated below.

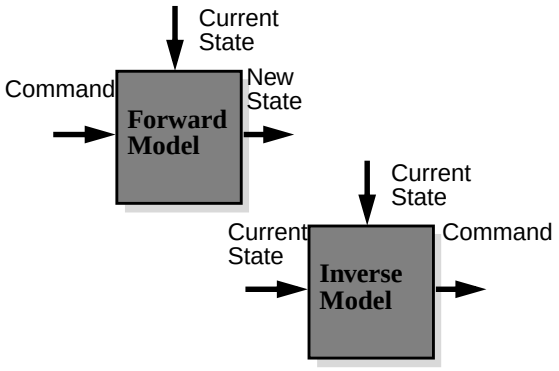


Figure 18: Forward and Inverse Models. These have analogous definitions in robot manipulator dynamics.

5.2.2 Subproblems

5.2.2.1 Trajectory Generation Problem

The command following problem arises either because the command itself was infeasible to begin with, or the controller performance is inadequate. If we choose to blame the trajectory rather than the controller, we will call the predicament the **trajectory generation problem**.

Legitimate constraints are imposed on trajectories by vehicle limitations that amount to very strong constraints on the feasibility of arbitrary trajectories expressed in configuration space. Such constraints include:

- actuator and plant kinematic and dynamic limits in the form of braking and steering maneuverability.
- underactuation of the vehicle
- processing and communication delays

Consider a simple situation where the vehicle command space consists of speed $V(t)$, and curvature $\kappa(t)$, and the configuration space consists of position $x(t), y(t)$ and heading $\psi(t)$ in two dimensions.

In attempting to generate trajectories that are feasible, several difficulties will emerge because the relation-

ship between a C-space trajectory and its command space equivalent is multivariate, nonlinear, coupled, and underdetermined.

The equations which map command space to C-space in a flat 2D world are given below:

$$\frac{dx(t)}{dt} = V(t)\cos\psi(t) \qquad x(t) = x_0 + \int_0^t V(t)\cos(\psi(t))dt$$

$$\frac{dy(t)}{dt} = V(t)\sin\psi(t) \qquad y(t) = y_0 + \int_0^t V(t)\sin(\psi(t))dt$$

$$\frac{d\psi(t)}{dt} = V(t)\kappa(t) \qquad \psi(t) = \psi_0 + \int_0^t V(t)\kappa(t)dt$$

Figure 19: Mapping from Command Space to C space. These are also the equations of dead reckoning.

We will call these equations a **forward model**. By contrast, an inverse model would be a solution to the problem of computing the A space curve from the C space curve. That is, the inverse model generates **clothoid** curves for Ackerman steered vehicles.

The generation of clothoids is a difficult problem that is further aggravated by the need to account for dynamics and latencies. Note however, that the forward model provides the inverse correspondence trivially, and in its integral form, is simply the equations of dead reckoning. The trajectory generation problem here is only difficult in one direction.

We have

5.2.2.2 Vehicle Model Fidelity Problem

It is often necessary for the system to understand the degree to which a particular motion can be accomplished. Poor fidelity of this aspect of the vehicle model means that the system will not understand its own motion. This in turn will lead to:

- unreliablility of obstacle avoidance
- instability of path following

The perceptual horizon of any vehicle is the maximum range of the perception sensor. At moderate speeds (<10 mph), it tends to be roughly equal to the worst case distance it takes for the steering actuator to move to its commanded position.

When dramatic steering changes are required to avoid a hazard, the navigation system operates almost entirely in the regime where curvature is continuously changing. This observation leads to the conclusion that the use of arc rather than clothoid models of Ackerman steering are incorrect at even moderate speeds.

The following figure indicates accurately the difference between an arc model and a clothoid model of vehicle response to a command to switch from a hard left to a hard right turn. The maximum rate of the steering wheel is 30° per second.

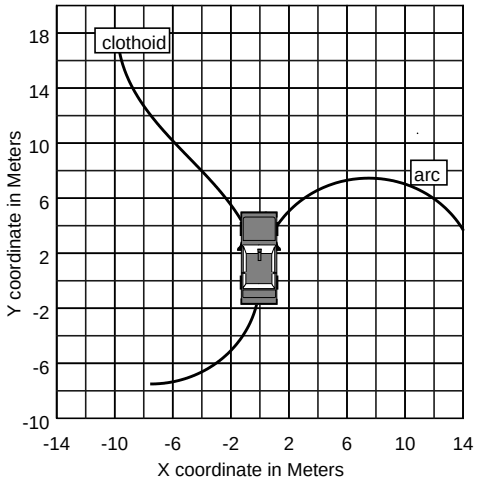


Figure 20: Model Fidelity at 5 m/s speed: The clothoid model correctly accounts for how long it really takes to reverse curvature.

Suppose an obstacle existed to the left of the vehicle and it issued a hard right command to avoid it. The response to such a command is the clothoid shown, so the vehicle would drive straight into the obstacle - being fundamentally unable to avoid it at the current speed.

5.2.3 Design

Our approach to managing these problems is to:

- Invert the order in which the common planning constraints of feasibility and admissibility are satisfied through a forward modeling approach to trajectory generation.
- Employ an accurate state space model of vehicle response as the forward model.

It turns out that it is far more efficient to search for an admissible trajectory in a space of feasible ones than vice-versa and that sufficiently accurate models of vehicle motion are relatively easy to generate in state space form.

5.2.4 Implications

An important implication of the approach is that the *generated trajectories are mechanically feasible by construction*. While the input commands to the model respect only the maximum curvature constraint, the output state estimate is consistent with the response of all actuators and the body kinematics of motion over rough terrain.

Thus, we have simplified the computation of the correspondence between command and response. This simple correspondence available through our control formulation combined with high fidelity models introduces the following benefits:

- **Reliability**: System reliability is enhanced

because dynamic feasibility is inherent in forward modeling approaches.

- **Accuracy**: Higher fidelity models make it possible to drive close to hazards when necessary and to track paths with low error.
- **Stability**: Vehicle control, whether for the purpose of obstacle avoidance or goal-seeking, remains stable at high speeds.
- **Performance**: Computational complexity of planning is reduced because the dynamics constraint is a valuable heuristic to limit search. This reduction in complexity leads to enhanced response times and higher speed motion.

5.3 Trajectory Representation & Generation

We will represent response trajectories implicitly in terms of the commands to which they correspond. When necessary, we will use a forward model to generate the response from the command through a process classically called **feedforward**.

The process which converts a command space trajectory to a state space trajectory is the solution of a constrained multidimensional differential equation which we will call the **state space model**.

5.3.1 Linear State Space Model

For a ***linear system***, the conventional state space model of a system is the following two matrix equations:

$$\frac{d\underline{\mathbf{x}}}{dt} = \mathbf{A}\,\underline{\mathbf{x}} + \mathbf{B}\,\underline{\mathbf{u}}$$

$$\underline{\mathbf{y}} = \mathbf{C}\,\underline{\mathbf{x}} + \mathbf{D}\,\underline{\mathbf{u}}$$

Note in particular that *the first equation is a differential one*. This kind of model is known classically as a **multivariate state space system**. It can be mapped onto our problem as follows. Let us assume the system is linear and describe the function of the matrices and vectors. The system of equations can be represented in a block diagram as follows:

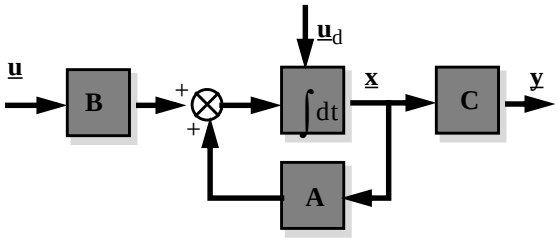


Figure 21: Multivariate Linear System Block Diagram. This model can be used to represent a vehicle driving over terrain.

The **system dynamics matrix**, **A**, models actuator constraints, kinematics, and dynamics, and body dynamics. It propagates the state of the vehicle forward in time. Our system model is based on the assumption that velocity can be considered constant for a small period of time.

The **input distribution matrix**, **B**, models communication delays and any kinematics that relate command space to actuation space[1].

The **command vector $\underline{\mathbf{u}}$** includes vehicle steering and speed commands as well as command signals to any articulated sensor heads. Generally, alternative commands can be any time-varying command vector $\underline{\mathbf{u}}(t)$.

The terrain disturbances $\underline{\mathbf{u}}_d$ model the terrain contact constraint[2]. Alternately, an abstract kinematic equation of the form $\underline{\mathbf{g}}(\underline{\mathbf{x}}) = \mathbf{0}$ can be used. Terrain geometry is represented in a terrain map data structure that is generated by the perception system.

The **state vector $\underline{\mathbf{x}}$** includes the vehicle steering, speed, and the state of motion of the vehicle body and any articulated sensor heads. It includes the 3D position and 3-axis orientation of the vehicle body as well as its linear and angular velocity.

The **output vector $\underline{\mathbf{y}}$** can be any function of both state and inputs. It will be discussed later in the context of obstacle avoidance.

5.3.2 Nonlinear State Space Model

The actual system model used is nonlinear. Fundamentally, this is because the actuators move with the vehicle, so the transformation from command space to state space depends on vehicle attitude and hence on state.

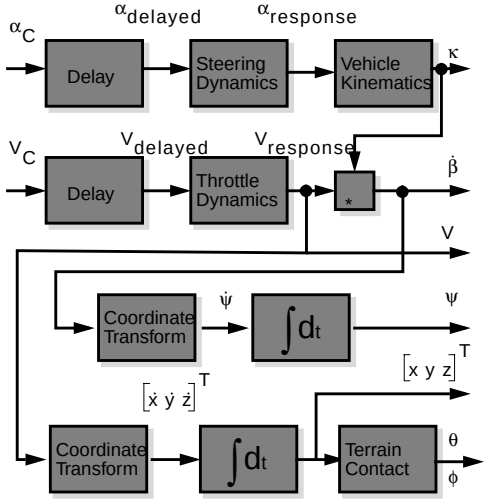


Figure 22: Forward model. This model is used to predict vehicle state given actuator commands (α_c, V_c). The output state contains rate of heading ($\dot{\beta}$), vehicle velocity(V), heading(ψ), pitch(θ), roll(θ) and position (x, y, z)).

1. For example, it would be the inverse Jacobian in resolved-rate control that maps cartesian inputs onto joint velocities.
2. Of course, at sufficiently high speeds, the vehicle need not remain in contact with the terrain.

The nonlinear model computes trajectories resulting from vehicle commands. The inputs to the model are the steering angle α_c, (corresponding to the desired path curvature), and throttle V_c, (corresponding to desired speed) and an elevation map of the terrain ahead of the vehicle.

The commands are first delayed through a FIFO queue to account for communications and processing and passed through a model of the actuator dynamics. In the case of the throttle (speed) the influence the gravitational load is so significant that it must be modeled.

The predicted steer angle response $\alpha_{response}$ is passed through a model of the steering column to predict the actual curvature κ, of the path traversed. The product of curvature and speed provides angular velocity $\dot{\beta}$. The linear velocity V is converted to world coordinates to generate the components of vehicle velocity along the world frame axes and then integrated to provide position (x, y, z).

Pitch θ and roll ϕ are determined by placing the vehicle wheels over the terrain map and allowing the vehicle to settle. Heading ψ is computed by integrating the angular velocity after converting coordinates to the world frame.

5.3.3 State Space Simulator

Thus, the basic simulation loop can be written as follows. At each time step:

- simulate suspension - determine attitude from terrain geometry and position
- simulate propulsion - determine new speed from command, state, and attitude
- simulate steering - determine angular velocity from steering and speed
- simulate body - dead reckon from linear and angular velocity and time step

The positions of distinguished points on the body, called **reference points**, are maintained in navigation coordinates throughout the simulation. The suspension model that is used is based on assumptions of rigid terrain and suspension and it computes the attitude of the vehicle which is consistent with terrain contact.

Propulsion is modeled as a proportional controller with gravity compensation. The steering model is based on an angular velocity limit on the steering wheel and the bicycle model of steering kinematics. Body dynamics are simulated using the 3D dead reckoning equations.

5.4 Trajectory Search

The basic search process used is generate and test. WE employ this technique while conducting a **command space search** over the feasible set of response trajectories.

The system considers a number of command space alternatives which span the entire set of commands available for the vehicle at some gross resolution. These are then converted to response state space and C space trajectories through feedforward and subsequently evaluated by both obstacle detection and goal seeking.

For a rigid-bodied vehicle moving in three dimensional space, the C-space can be considered to be a subset of state space - that is, the coordinates of the vehicle control point expressed as (x, y, z, roll, pitch, yaw). The command space for a conventional automobile is spanned by the variables of speed and path curvature and these variables map more or less directly to the controls of throttle and steering.

5.4.1 Feedforward

By the definition of state, the system can be projected arbitrarily far forward in time based only on the command signal, terrain contact constraint, and time. Hence, the system model constitutes a **state space simulator** as shown below.

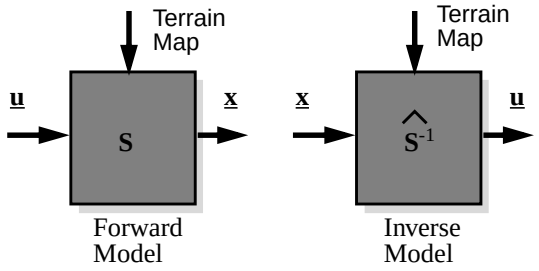


Figure 23: State Space Simulator. Because a state space model retains state, it can be used to project motion that corresponds to a command arbitrarily far into the future. The inverse model on the right is never explicitly evaluated.

The set of trajectories $\underline{\mathbf{x}}_i(t)$ which:

- satisfies the model equations ($d\underline{x}/dt = \mathbf{A}\ \underline{x} + \mathbf{B}\ \underline{u}$)
- maintains contact with rigid terrain ($\underline{g(x)} = \mathbf{0}$)

is called the **feasible set**.

The constraints are satisfied by construction through feedforward of the system dynamics and altering the vehicle attitude at each step in the simulation to enforce terrain contact.

5.4.2 Predictive Control vs. C Space Planning

The differences between classical C space planning and command space planning are indicated in the following figure. On the left of the figure, the search of planning alternatives is expressed in **configuration space**. A commonly invoked assumption is the expression of obstacles as discrete points in this space. When a clear region or set of points has been found in front of the vehicle, a **trajectory generation** algorithm is invoked to map C space onto the vehicle command

space and these commands are sent to the hardware for execution.

On the right of the figure is the **predictive control** approach. Note first that the direction of the arrows are reversed. The inverse system model is never evaluated explicitly. The system simply remembers the correspondence of command to response trajectories and inverts this list of ordered pairs.

It is clear from the figure that state space is, in fact, a superset of configuration space - including all C space variables plus any derivatives that appear in the system dynamic model.

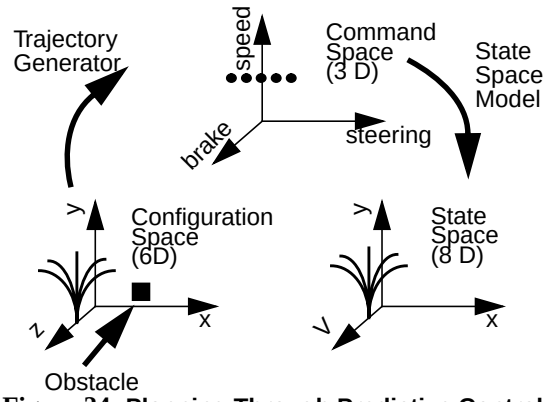


Figure 24: Planning Through Predictive Control. Obstacles are most naturally represented in C space but generating commands that avoid them involves a difficult inverse model. Instead, a representative number of command space alternatives are transformed, through feedforward, to state space, and then to C space and then checked for intersection with obstacles.

5.5 Results

While it is difficult to compute the shapes of regions in configuration space in closed form, it is relatively easy to write a computer program to enumerate all possibilities and fill in boxes in a discrete grid which represents C-space at reduced resolution. The three dimensional C-space for an Ackerman steer vehicle for an impulse turn at 4.5 m/s was generated by this forward technique.

The results are plotted below in heading slices of 1/16 of a revolution. Symmetry generates mirror images along the heading axis, so two slices are plotted on each graph. The maneuver is a turn from zero curvature to the maximum issued at time t = 0. A dot at a particular point (x,y) in any graph indicates that the heading of the slice is obtainable at that position. There are 16 slices in total of which 6 are completely empty (i.e the vehicle cannot turn around completely in 20 meters). The total percent occupancy of C-space is the ratio of the total occupied cells to the total number of cells. This can be computed from the figure to be 3.1%.

So *97% of the C-space of the vehicle is infeasible* if the limited maneuverability of the vehicle is modeled. The maneuverability is limited by both the nonzero minimum turn radius and the steering actuator response. Note that occupancy of C-space does not account for higher level dynamics. There are severe constraints on the ability to “connect the dots” in these graphs which aggravate the situation further.

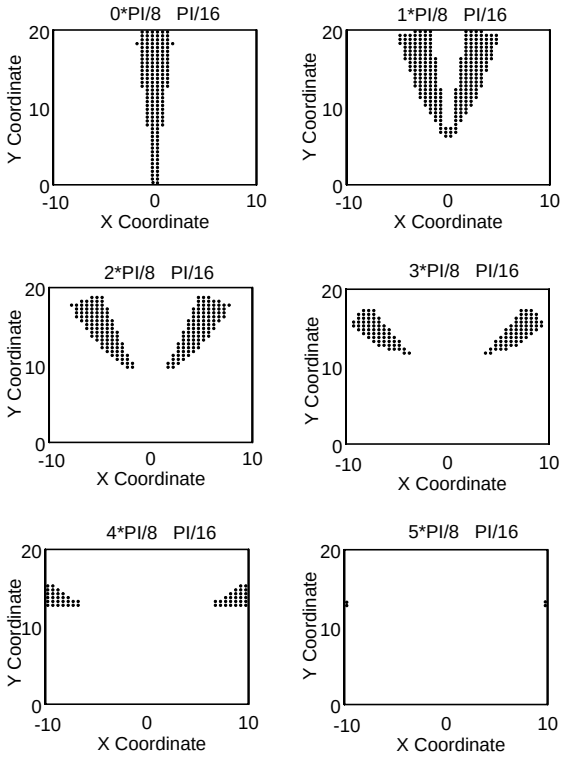


Figure 25: Ackerman Steer Configuration Space. For a sharp turn to the left or right at 4.5 m/s, 97% of the configuration points in front of the vehicle are not feasible.

6 Goal Arbitration

This section discusses the motivation behind, and the implementation of our optimal control approach to goal arbitration. The proposed approach is similar in formulation to work in classical optimal control [37] in that it seeks to determine control signals that will both satisfy constraints and optimize a performance criterion.

6.1 Introduction

In addition to trajectory search, the local intelligent mobility problem involves an aspect of goal arbitration. For example, given a goal path to follow and the simultaneous goal of avoiding obstacles, it is likely

and frequently the case that these goals will conflict. More plainly, an obstacle may appear directly on the goal path.

We will discuss here our mechanism for dealing with this conflict as well as the manner in which candidate trajectories are ranked for both their obstacle avoidance and goal-seeking potential.

6.1.1 Problem

Let us define the **strategic goal** as some path or position to be followed or achieved and the **tactical goal** as that of simultaneously avoiding all hazardous conditions. If both goals are implemented as independent behaviors they will naturally disagree on the commands to the actuators. This legitimate and inevitable conflict will be called the **actuator contention problem**.

6.1.2 Solution

Solutions to this problem must decide how to either:

- Merge both commands together to generate a third.
- Give one behavior priority over the other.

Regardless of how this is done, the general technique involved is **arbitration**. Several approaches have been used ranging from subsumption of one behavior in favor of another [8] to consensus-building and voting techniques [47].

A spectrum of approaches exist with extremes that correspond roughly to bureaucracy and democracy. Our approach is somewhat intermediate between these extremes. It recognizes that:

- Some behaviors, like obstacle avoidance, must be given absolute veto power over unsafe trajectories.
- Others, like goal seeking can profitably optimize performance through search and ranking of the remaining alternative trajectories.

6.2 Preliminaries

6.2.1 Terminology

Any number of potential hazardous situations may exist along a trajectory. Some of these hazards include:

- **discrete obstacles** like rocks, holes, and trees that would damage the vehicle through collision.
- **hazardous configurations** like extreme pitch and roll angles that would damage the vehicle through tipover.
- **hazardous states** like extreme lateral or longitudinal acceleration which could damage the vehicle through tipover or loss of traction and control.
- **traction traps** like wheel sized holes, high centering terrain, and regions of ice or slippery terrain that would prevent further vehicle motion.
- **catastrophic falls** like ravines and cliffs that could damage the vehicle through complete loss of terrain contact, followed by ballistic motion, followed by catastrophic collision.

A path is **admissible** if it would be safe for the vehicle to traverse it.

6.2.2 Design

Our basic approach is to notice that the mobility problem can be expressed in the familiar terms of optimal control theory and to then apply the associated techniques and abstractions of this theory.

Architecturally, the tactical control layer consists of coexisting hazard avoidance and goal seeking behaviors that are managed by an arbiter to avoid conflict as shown below:

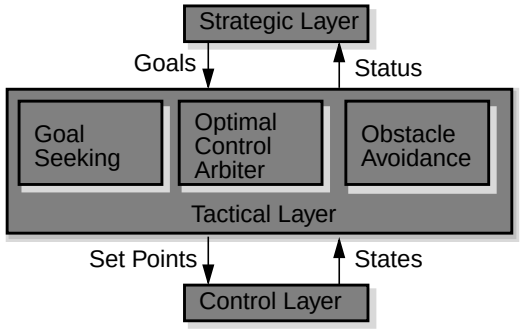


Figure 26: Optimal Control Arbitration. The tactical control layer consists of a trajectory generator (not shown), two purposeful behaviors, and an arbiter to coordinate them.

In fact, the seeking and/or avoidance occur when the arbiter chooses an alternative and sends it to the control layer. The other two entities simply rank the candidate trajectories that are given to them.

6.2.3 Implications

Our approach has the advantage that some limited degree of local intelligent behavior emerges naturally. For example, wall or other extended feature following emerges because optimization will keep the vehicle close to an obstacle between it and the goal. However, once a break in the extended feature appears, the system will immediately seize the opportunity to reacquire the goal path.

6.3 Arbitration

Note that the satisfaction of either goal may be expressed as a constraint or as some utility function to be optimized. For example, we could optimize safety by choosing the safest overall candidate trajectory or we might rigidly enforce a path to follow as a constraint. We believe the reverse is more appropriate. That is, hazard avoidance is a constraint and goal seeking is to be optimized.

Hence, the task of safe navigation can be expressed in classical optimization terms. The navigator must

achieve some useful goal while simultaneously satisfying the constraint of avoiding damage to the vehicle or its payload and the constraints of limited vehicle maneuverability.

6.3.1 Feasible Set

We will express this optimal control problem as follows. Let $\underline{\mathbf{u}}_i(t)$ be a candidate command signal. The response to this candidate command, denoted $\underline{\mathbf{x}}_i(t)$, is generated from the nonlinear system dynamics and terrain following models:

$$\dot{\underline{\mathbf{x}}} = \mathbf{f}(\underline{\mathbf{x}},\underline{\mathbf{u}}) \qquad \text{System Dynamics}$$

$$\mathbf{g}(\underline{\mathbf{x}}) = \underline{\mathbf{0}} \qquad \text{Terrain Following}$$

Let the set of mechanically feasible trajectories, denoted $\underline{\mathbf{X}}_f$, be those which satisfy the above two equations. The response trajectory is a member of this **feasible set**:

$$\underline{\mathbf{x}}_i(t) \in \underline{\mathbf{X}}_f \qquad \text{Feasibility}$$

Note that the space of possible commands to issue is continuous. Rather than deal with variational calculus, we will sample the feasible set of trajectories at some practical resolution that ensures adequate coverage of the set and search the alternatives exhaustively.

6.3.2 Output Vector

The **hazard vector**, $\underline{\mathbf{y}}$, consists of rankings of every point along a response trajectory for its relative safety in terms of several of the hazard conditions mentioned earlier. By and large, safety can be determined kinematically from the state and the terrain map. Each element of the vector corresponds to a different hazard.

$$\underline{\mathbf{y}} = \mathbf{h}(\underline{\mathbf{x}}) \qquad \text{Hazard Kinematics}$$

Let us define the safety threshold vector, $\underline{\mathbf{y}}_{\mathbf{safe}}$, as the constant vector whose elements are the maximum safe values of each element of the hazard vector. A trajectory is safe when:

$$|\underline{\mathbf{y}}| < \underline{\mathbf{y}}_{\mathbf{safe}} \qquad \text{Safety}$$

Let the set of safe trajectories, denoted $\underline{\mathbf{X}}_a$, be those whose associated hazard vectors satisfy the above equation. Such a response trajectory is a member of the **admissible set**:

$$\underline{\mathbf{x}}_i(t) \in \underline{\mathbf{X}}_a \qquad \text{Admissibility}$$

6.3.3 Optimal Control Problem

While there are many potential forms of strategic goal that might be assigned to the vehicle, let us assume, without loss of generality, that a **goal trajectory**, $\underline{\mathbf{x}}_{\mathbf{goal}}$, of some form has been assigned.

Further, let us define a **goal functional, $\underline{\mathbf{L}}[\underline{\mathbf{x}}_i(t)]$**, to be an arbitrary expression of how well a particular trajectory follows the goal trajectory. If we use the integrated length of the vector difference, we might write:

$$\underline{\mathbf{L}}[\underline{\mathbf{x}}_i(t)] = \|\underline{\mathbf{x}}_i(t) - \underline{\mathbf{x}}_{\mathbf{goal}}(t)\|$$

The optimal control problem can now be represented as follows:

$$\underset{\underline{\mathbf{u}}_i(t)}{\text{Minimize:}}\ \underline{\mathbf{L}}[\underline{\mathbf{x}}_i(t)] = \|\underline{\mathbf{x}}_i(t) - \underline{\mathbf{x}}_{\mathbf{goal}}(t)\| \qquad \text{Goal Proximity}$$

$$\text{Subject to:}\quad \dot{\underline{\mathbf{x}}} = \mathbf{f}(\underline{\mathbf{x}},\underline{\mathbf{u}}) \qquad \text{System Dynamics}$$

$$\mathbf{g}(\underline{\mathbf{x}}) = \underline{\mathbf{0}} \qquad \text{Terrain Following}$$

$$\underline{\mathbf{y}} = \mathbf{h}(\underline{\mathbf{x}}) \qquad \text{Hazard Kinematics}$$

$$|\underline{\mathbf{y}}| < \underline{\mathbf{y}}_{\mathbf{safe}} \qquad \text{Safety Constraint}$$

Figure 27: Optimal Control Arbitration. Driving safely toward a goal can be formulated in terms of optimizing a functional over a set of trajectories that are both safe and mechanically feasible.

6.4 Adaptive Regard

Although predictive control manages the complexity of trajectory search, it does nothing to minimize the cost of evaluating a particular trajectory for its safety and/or goal seeking potential. Efficient trajectory evaluation is the subject of this section.

We minimize wasted computation in trajectory evaluation by selectively processing only the data that matters along the trajectory in the environmental model. Known here as **adaptive regard**, the technique is the analog of our adaptive approach to perception in that it minimizes references to the environmental model just as adaptive perception minimizes references to images.

6.4.1 Detection Zone

The immediately material information forms a region in the environmental model which we call the **detection zone** - that region of the near environment which the vehicle can reach but is not already committed to travelling and about which an immediate decision of

traversability is needed to continue moving.

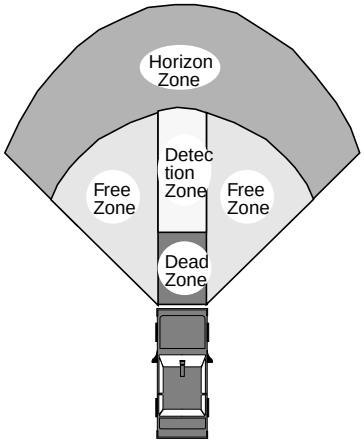


Figure 28: Detection Zone For Braking Maneuver. In any candidate vehicle maneuver a swath of ground is covered. The detection zone is the region which must be verified to be safe in the current computational cycle to prevent collisions.

6.4.2 Remaining Zones

Thus, in our search for hazards we *do not* look for hazards in the following regions:

- **free zone**: where the vehicle cannot go
- **dead zone**: where the vehicle is committed to go
- **horizon zone**: where the decision can be delayed

The precise location of the detection zone is obtained trivially from a time window into the response trajectories computed by command feedforward in the state space model.

6.4.3 Planning Window

The **planning window** planning is analogous to the **range window** in perception. It confines the search for hazards to the detection zone. One of the highest level system requirements is to attempt to maintain continuous motion, so adaptive regard is based on turning maneuvers (which consume more space) rather than braking maneuvers.

An **impulse turn** is a turn from zero curvature to the maximum allowed curvature. The planning window is computed by predicting the distance required to execute an impulse turn at the current speed with the best available estimates of the output latencies that will apply.

Precision in computation of the planning window requires careful treatment of time. The planning window is measured from the position where the vehicle will be when the steering actuator starts moving.

6.4.4 Real-Time Latency Modeling

The planning problem has latency concerns similar to those of perception. Without latency models, obstacle avoidance significantly underestimates the distance required to react for two reasons.

- Position estimate input latency implies that the vehicle is actually much closer to an obstacle than the last available position measurement suggests.
- Command output latency and actuator dynamics imply that it actually takes much more distance to turn than would be expected from instantaneous response models.

A model of these latencies is accomplished with the following mechanisms:

- all input and output is time-stamped
- all input and output is stored in internal FIFO queues
- all sensor latencies are modeled
- all actuator latencies are modeled

These FIFO queues do not introduce artificial delay - they are used to *model* the delays which already exist in the system.

6.5 Hazard Detection

The process of hazard detection is the process of computing the hazard output vector. The hazard vector moves over time in a **hazard space** in response to the movement over time of the vehicle state vector in state space. That is, there is a hazard trajectory $\underline{\mathbf{y}}_i(t)$ which corresponds to each state trajectory $\underline{\mathbf{x}}_i(t)$, which in turn corresponds to each command signal vector, $\underline{\mathbf{u}}_i(t)$.

Once the vehicle trajectory $\underline{\mathbf{x}}_i(t)$ is known, a set of hazard models is used to compute its relative safety with respect to the associated terrain information in the environmental model.

6.5.1 Types of Hazards

Each element of the hazard vector corresponds to a different hazardous condition. Each scalar hazard $\mathbf{y}_j(t)$ is a function of the vehicle state, the terrain on which it rests, and the input commands. Some typical hazards are:

- **Tipover**: The movement of the weight vector outside of the support polygon formed from the wheel contact points.
- **Body Collision**: Collision of the underbody with the terrain.
- **Discrete Obstacles**: Regions of locally high terrain gradient.
- **Unknown Terrain**: Regions that are occluded, outside the field of view of the sensors, unknown from poor measurement accuracy or resolution, or devoid of matter (such as the region beyond a cliff edge).

6.5.2 Hazard Signals

As an example of a hazard signal, consider expressing proximity to tipover in terms of excessive pitch or roll angle. Let the elevations of the front and rear points on

the vehicle longitudinal axes after enforcing terrain contact be $z_f(t)$ and $z_r(t)$. The pitch tipover hazard signal is then given by:

$$y_{pitch}(t) = |z_f(t) - z_r(t)|/L$$

The other three hazards above are computed from the volume under the belly, the local terrain gradient, and the total trajectory length for which the terrain is unknown respectively.

6.5.3 Trajectory Safety

In order to assess the safety of an entire trajectory, it is necessary to integrate out the time dimension and merge all of the predictions of different hazard types together. This process generates a holistic estimate of the degree of safety expected if the associated command is executed.

Although many alternatives for doing this present themselves, we have achieved acceptable performance by ranking the whole trajectory using the worst hazard at the worst point in time. The output of this process is the safety ratings of all trajectories - which is supplied later to the optimal control arbiter.

6.6 Goal Seeking

The process of goal-seeking starts with the process of computing the goal functional. Many techniques for computing the proximity of two trajectories are possible, but the one we chose here is a modification of the classical **pure pursuit** algorithm.

The most general form of path-based strategic goal is a literal trajectory to follow[1]. Other types of goals such as headings, points, and curvatures, can be extracted as trivial subcases of the more general path tracking problem.

6.6.1 Basic Pure Pursuit

The pure pursuit algorithm [48] is a proportional controller formed on the heading error computed from the current heading and the heading derived from the current vehicle position and a **goal point** on the path.

The goal point is computed by finding the point on the path which is a predetermined distance L from the current vehicle position.

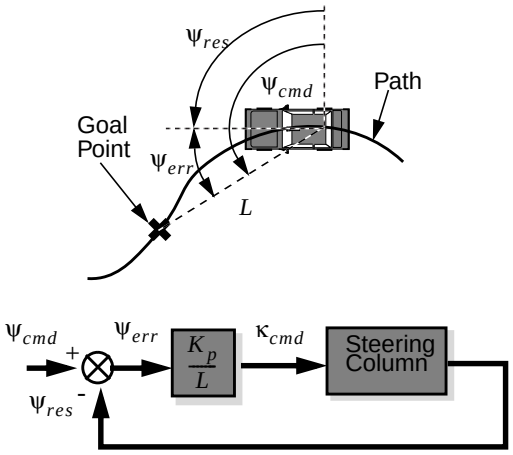


Figure 29: Basic Pure Pursuit. A proportional controller formed on the heading error to acquire a goal point at a given lookahead distance.

Heading is measured at the center of the rear axle. The proportional gain K_p is normalized by the lookahead distance L. This can be viewed an adaptive element or, more simply, as a unit conversion from heading error to curvature because the ratio ψ_{err}/L is the average curvature required to reacquire the path at the goal point.

6.6.2 Rough Terrain Pure Pursuit

A few modifications are introduced to adapt pure pursuit for rough terrain. Extremely large tracking errors must be acceptable to the servo without causing instability if obstacles are to be avoided robustly. This is made possible by two devices indicated in the following figure.

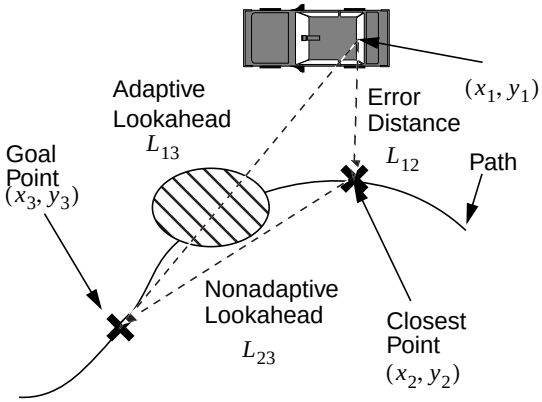


Figure 30: Adaptive Pure Pursuit. The point closest to the vehicle on the path replaces the vehicle position.

The system maintains a running estimate of the point on the path which is closest to the current vehicle position in order to avoid an expensive search of the entire path each iteration. This closest point is used instead of the current vehicle position as the origin of the loo-

1. The arbiter can easily integrate the real-time inputs of a human supervisor through this mechanism.

kahead vector.

The lookahead distance is adaptive to the current tracking error - increasing as the error increases as indicated in the accompanying code fragment:

```
min = HUGE; way_pt = close;
while(L23 < L12 + lookahead)
   {
   x3 = path_x[way_pt];
   y3 = path_y[way_pt];
   if( L13 < min )
      {
      close = way_pt;
      }
   way_pt++;
   }
way_pt = close;
while(L13 < L12 + lookahead)
   {
   x3 = path_x[way_pt];
   y3 = path_y[way_pt];
   goal_pt = way_pt;
   }
```

Figure 31: Adaptive Pure Pursuit Algorithm. The lookahead is adapted to allow for large tracking errors when obstacles are avoided.

The first while loop is responsible for maintaining a running record of the close point, point 2. It searches through an arc length window which adapts to the path tracking error. As the error gets larger, this loop will cause the close point to jump over high curvature kinks in the path as they become less relevant at the resolution of the tracking error.

The second while loop computes the goal point in an identical manner. It basically moves point 3 forward until it falls outside a circle centered at the vehicle whose radius is the sum of the error distance L_{12} and the nonadaptive lookahead L_{23}.

In this way, when an obstacle avoidance maneuver causes significant path error, the algorithm will search to reacquire the path on the other side of the obstacle instead of causing a return to the front of the obstacle.

Under normal circumstances when the vehicle is tracking the path, the close point is at the vehicle position, the error distance is close to zero, and the adaptive lookahead is the nonadaptive lookahead. Hence, the algorithm gracefully degenerates to classical pure pursuit when obstacle avoidance is not necessary.

6.6.3 Feedforward Pure Pursuit

The devices of the previous section account for obstacle avoidance. However, basic pure pursuit also suffers from speed related problems. Instability results from large tracking errors, too short a lookahead distance or too high a gain.

The goal functional is generated by evaluating, at each point on each candidate trajectory, the distance from the vehicle to the goal point. The minimum distance over the entire trajectory is the functional value associated with it.

The following figure shows how accurate models of response stabilizes goal seeking.

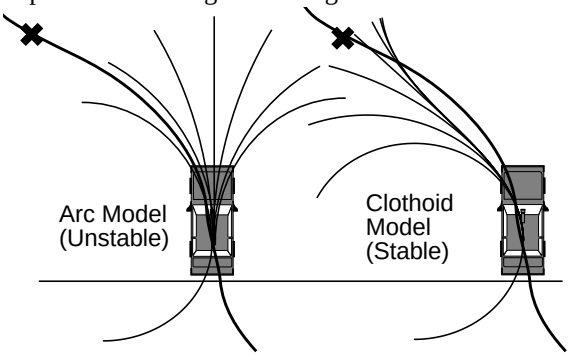


Figure 32: Feedforward Pure Pursuit. Inaccurate models of response lead to servo instability. Note that the vehicle is in a sharp left turn and happens to have zero heading at t=0.

In the above situation, if an arc-based model were used, the system would issue a hard left command. However, the more accurate clothoid model reveals that such a command would actually *increase* the tracking error leading to even more overcorrection. A tracker based on a more accurate clothoid model would recognize the situation and issue a zero curvature command that would correctly acquire the goal at the lookahead distance.

6.7 Results

In the RANGER navigator, command alternatives are expressed in terms of constant speed, constant curvature commands. Several times a second, the optimal control arbiter considers approximately ten steering angles to use during the next control cycle. The forward model simulates the effect of using these steering angles over a short period of time and evaluates each of the resultant paths. Any steering angles that result in paths that go near or through hazardous vehicle configurations are discarded. The steering angle that results in a path that is optimal based on several criteria is chosen.

In the figure below the system issues a left turn command to avoid a hill to its right. The histograms represent the votes for each candidate trajectory (higher values indicate safer trajectories). The hazards are:

- ROLL: excessive roll
- PITCH: excessive pitch
- BODY: collision with the body

• WHEEL: collision with the wheels

The tactical vote (TACT) is the overall vote of hazard avoidance. The strategic vote (STRAT) is the goal seeking vote.The arbiter chooses the third trajectory from the left because this closest to the strategic vote maxim (straight) while exceeding the threshold for safety.

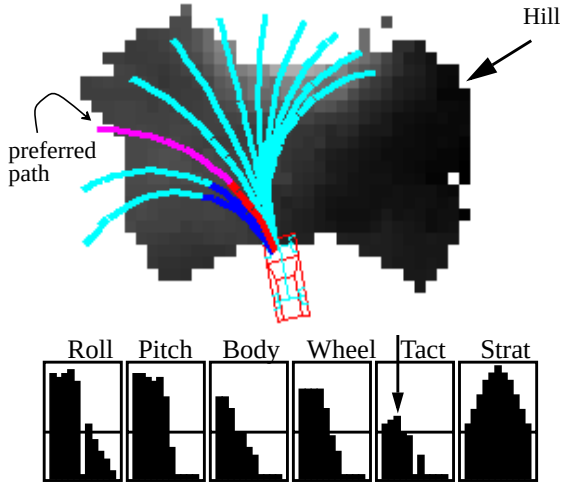


Figure 33: Optimal Control Arbitration.The system chooses a steering angle from a set of candidate trajectories.The histograms represent “votes” for each candidate trajectory where higher values indicate preferred trajectories.

7 Summary and Conclusions

This section summarizes our perspectives and conclusions.

7.1 Perspectives

Our implementation of a tactical control layer implies a perspective on the need for deliberation and reactivity in autonomous vehicles. It seems that both reactivity and deliberation have their place in our approach to local intelligent mobility.

7.1.1 Memory

From a real-time response point of view, high-speed navigators cannot afford the computation necessary to continually process the same scene geometry at sufficiently high resolution. Thus, for high-speed navigators, the memory involved in the mapping of the environment is an essential system capability.

7.1.2 Deliberation

For high-speed navigators, models of dynamics take the place of models of logical precedence used in AI in that they limit the states reachable in a small period of time from any given state. In such navigators, the deliberative reasoning about the future impact of current actions that is implemented in feedforward models is an essential capability - at least to the extent that the system must understand its own motion.

7.1.3 Reaction

The latency models developed in the paper and the precision timekeeping that has been incorporated in the software seem more applicable to the control systems of fighter aircraft than to autonomous vehicles. It seems that high-speed navigators must *reason about* their ability to respond.

7.2 Conclusions

This section presents a short list of conclusions which seem most significant to the problem and most relevant to the more general problem of autonomous mobility.

7.2.1 Tactical Control Layer

A modification on a standard architectural model of robotic systems has been proposed which connects strategic geometric reasoning to dynamic reactive control in an effective manner when the system under control exhibits poor command following. The problem is solved in this intermediate layer between AI and control, between reactive and deliberative approaches.

7.2.2 Adaptive Perception

The throughput problem of autonomous navigation can be managed at contemporary speeds by computational stabilization of the sensor sweep and active control of resolution through intelligent image subsampling.

7.2.3 Computational Image Stabilization

Adaptive perception techniques which computationally stabilize the vertical field of view provide the best of both worlds. They provide the high throughput necessary for high-speed motion and the wide field of view necessary for rough terrain.

7.2.4 Adaptive Regard

In a manner similar to the use of a focus of attention in perception, a focus of attention can be computed for obstacle avoidance that reflects the capacity of the vehicle to react at any given time. **Adaptive regard** places a limit on how close a vehicle should look for hazards because it cannot react inside of some distance. Thus, adaptive regard calls for all data inside some lower limit to be ignored and limits are placed on the extent of data processed beyond this minimum limit by considerations of both minimum planning throughput and range data quality.

7.2.5 Command Space Search

Dynamics of many kinds imply that the local planning

problem is actually relatively easy from the point of view of search complexity. The planning "state space[1]" of the high-speed Ackerman vehicle is degenerate. Ordering heuristics generally optimize search by imposing the most constraining limits first and reducing the size of the search space as fast as possible. This principle is used here because once dynamically infeasible paths are eliminated, only a few remaining alternatives are spatially distinct enough to warrant consideration.

7.2.6 Dynamic Models

The incorporation of dynamics models has generated a local navigation system which remains stable past the limits beyond which our kinematically modeled systems became unstable. The use of dynamic models also makes obstacle avoidance more reliable in general by imparting to the system a more accurate understanding of its ability to respond.

7.2.7 Forward Modeling

Physical dynamics amounts to an overwhelming constraint on the maneuverability of a high-speed vehicle. For a vehicle or operating regime for which classical path generation based on via points becomes difficult, forward modeling has the advantage that generated trajectories are feasible by construction.

7.2.8 Arbitration

The simultaneous satisfaction of hazard avoidance and goal seeking can cause contention for the absolute control of vehicle actuators, and in a practical system, this contention must be resolved through some arbitration mechanism. The problems of goal seeking, local path planning, and hazard avoidance have been unified into an optimal control context. In this context, a functional computed over the feasible set of response trajectories serves as the quantity to be optimized and the hazard avoidance mechanism specifies the confines of the feasible set.

7.2.9 Hazard Space

In strict mathematical terms, the configuration space (C-space) of AI planning is a subset of the state space (S-space) of multivariate control. It is a well-established technique to abstract a mobile robot into a point in six-dimensional position and attitude coordinates. The significance of hazard space (H-space) is that it performs the same function for dynamic planning that C-space performs for kinematic planning.

8 Acknowledgements

This research was sponsored by ARPA under contracts "Perception for Outdoor Navigation" (contract number DACA76-89-C-0014, monitored by the US Army Topographic Engineering Center) and "Unmanned Ground Vehicle System" (contract number DAAE07-90-C-RO59, monitored by TACOM).

1. In AI, the term "state space" has a slightly different connotation than in control theory.

9 References

1. J. Aliomonos, I. Weiss, and A. Badyopadhyay. Active Vision, Int. J. Computer Vision, 1, pp. 333-356, January 1988.
2. R Bajcsy, "Active Perception", Proc. IEEE, 76(8), pp. 996-1005, August 1988.
3. M. G. Bekker, "Off-the-Road Locomotion", The University of Michigan Press, 1960.
4. M. G. Bekker, "The Theory of Land Locomotion", The University of Michigan Press, 1956.
5. R. Bhatt, L. Venetsky, D. Gaw, D. Lowing, A. Meystel, "A Real-Time Pilot for an Autonomous Robot", Proceedings of IEEE Conference on Intelligent Control, 1987.
6. B. Blochl, "System Requirements and architectures for vision-guided autonomous robots", in Proceedings of the 12th European meeting on Cybernetics and Systems Research, Vienna, 5-8 April, 1994.
7. B. Brumitt, R. C. Coulter, A. Stent, "Dynamic Trajectory Planning for a Cross-Country Navigator", Proceedings of the SPIE Conference on Mobile Robots, 1992.
8. R. Brooks, "A Hardware-Retargettable Distributed Layered Architecture for Mobile Robot Control", Proceedings of the IEEE International Conference on Robotics and Automation, 1987.
9. T. S. Chang, K. Qui, and J. J. Nitao, "An Obstacle Avoidance Algorithm for an Autonomous Land Vehicle", Proceedings of the 1986 SPIE Conference on Mobile Robots, pp. 117-123.
10. M. Daily et al., "Autonomous Cross Country Navigation with the ALV", In Proc. of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, Pa, April 1988; pp. 718-726.
11. E. D. Dickmanns. "Dynamic Computer Vision for Mobile Robot Control", Proceedings of the 19th International Symposium and Exposition on Robots, pp. 314-27;
12. E. D. Dickmanns, A. Zapp, "A Curvature-Based Scheme for Improving Road Vehicle Guidance by Computer Vision", Proceedings of the SPIE Conference on Mobile Robots, 1986.
13. R. T. Dunlay and D. G. Morgenthaler, "Obstacle Detection and Avoidance from Range Data", In Proc. SPIE Mobile Robots Conference, Cambridge, Mass., 1986
14. R. T. Dunlay, D. G. Morgenthaler, "Obstacle Avoidance on Roadways Using Range Data", Proceedings of SPIE Conference on Mobile Robots, 1986.
15. Feiten, W. and Bauer, R., "Robust Obstacle Avoidance in Unknown & Cramped Environments," In Proc. IEEE International. Conference on Robotics and Automation, May 1994. San Diego.
16. D. Feng, S. Singh, B. Krogh, "Implementation of Dynamic Obstacle Avoidance on the CMU Navlab", Proceedings of IEEE Conference on Systems Engineering, August, 1990.
17. R. Fikes and N. Nilsson, "STRIPS: A New Approach to the

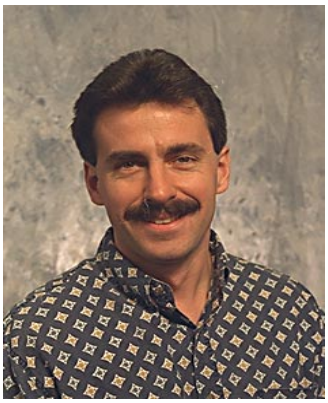
Application of Theorem Proving to Problem Solving", Artificial intelligence, Vol 2., pp. 189-208 (1971).

18. U. Franke, H. Fritz, S. Mehring, "Long Distance Driving with the Daimler-Benz Autonomous Vehicle VITA", PROMETHEUS Workshop, Grenoble, December, 1991.
19. M. Georgeff and A. Lansky, Reasoning about actions and plans: Proc. of the 1986 workshop,1986, AAAI.
20. J. Gowdy, A. Stentz, and M. Hebert, "Hierarchical Terrain Representation for Off-Road Navigation", In Proc SPIE Mobile Robots, 1990.
21. P. Grandjean and L. Matthies, "Perception Control for obstacle detection by a cross-country rover", in Proceedings of the IEEE International Conference on Robotics and Automation, May 1993.
22. M. Hebert, T. Kanade, and I. Kweon. "3-D Vision Techniques for Autonomous Vehicles", Technical Report CMU-RI-TR-88-12, The Robotics Institute, Carnegie Mellon University, 1988
23. M. Hebert and E. Krotkov, "Imaging Laser Radars: How Good Are They", IROS 91, November 91.
24. Hebert, M., Pomerleau, D., Stentz, A., Thorpe, C., "A Behavior-Based Approach to Autonomous Navigation Systems: The CMU UGV Project", to appear in IEEE Expert.
25. J. Hendler, A. Tate, and M. Drummond, "AI Planning: Systems and Techniques," AI Magazine, Summer, 1990, Vol 11 (2).
26. R. Hoffman, and E. Krotkov, "Terrain Mapping for Long Duration Autonomous Walking", In Proc. IEEE/RSJ International. Conference on Intelligent Robots and Systems, Raleigh, 1992.
27. B. K. Horn and J. G. Harris, "Rigid Body Motion from Range Image Sequences", Image Understanding, Vol. 53, No. 1, January 1991, pp 1-13.
28. B. Hotz, Z. Zhang, P. Fua, "Incremental Construction of Local DEM for an Autonomous Planetary Rover", Workshop on Computer Vision for Space Applications - Antibes, Sept. 22-24, 1993.
29. M. Jordan and R. Jacobs, "Learning to Control an Unstable System with Forward Modeling," In Advances in Neural Information Sciences 2, Morgan Kaufmann, 1990.
30. A. Kelly, A. Stentz, M. Hebert, "Terrain Map Building for Fast Navigation on Rough Terrain", Proceedings of the SPIE Conference on Mobile Robots, 1992.
31. A. Kelly, "An Intelligent Predictive Control Approach to the High-Speed, Cross Country Autonomous Navigation Problem", Ph. D. thesis, Robotics Institute, Carnegie Mellon University, June 1995.
32. A. Kelly, and A. Stentz, "Analysis of Requirements of Rough Terrain Autonomous Mobility: Part I: Throughput and Response", IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, April 1997.
33. A. Kelly, and A. Stentz, "Analysis of Requirements of Rough Terrain Autonomous Mobility: Part II: Resolution and Accuracy", IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, April 1997.
34. A. Kelly, and A. Stentz, "Computational Complexity of Terrain Mapping Perception in Autonomous Mobility", IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, April 1997.
35. A. Kelly, and A. Stentz, "Analysis of Requirements of Rough Terrain Autonomous Mobility", submitted to Autonomous Robots.
36. D. Keirsey, D. Payton, J. Rosenblatt, "Autonomous Navigation in Cross-Country Terrain", proceedings of Image Understanding Workshop, 1988.
37. D. E. Kirk, Optimal Control Theory, Prentice Hall, 1970.
38. E. Krotkov, and M. Hebert, "Mapping and Positioning for a Prototype Lunar Rover", proceedings of the International Conference on Robotics and Automation, May 1995.
39. In So Kweon, "Modelling Rugged Terrain by Mobile Robots with Multiple Sensors", CMU PhD Thesis, 1990
40. T. Lozano-Perez, and M. A. Wesley, "An Algorithm for Planning Collision Free Paths Among Polyhedral Obstacles", Communications of the ACM, Vol. 22, Num 10, October 1979, pp. 560-570.
41. L. Matthies, "Stereo Vision for Planetary Rovers", International Journal of Computer Vision, 8:1, 71-91, 1992.
42. L. Matthies, and P. Grandjean, "Stochasitc performance modelling and evaluation of obstacle detectability with imaging range sensors." IEEE Transactions on Robotics and Automation, Vol 10, Number 6, December 1994, pp 783-791
43. L. Matthies, A. Kelly, T. Litwin, "Obstacle Detection for Unmanned Ground Vehicles: A Progress Report," to appear in the Proceedings of the IEEE International Symposium on Intelligent Vehicles.
44. H. P. Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover", Ph. D. Thesis, Stanford University, 1980.
45. K. Olin and D. Tseng. "Autonomous Cross Country Navigation", IEEE Expert, August 1991, pp. 16-30.
46. K. M. Passino and P. J. Antsaklis, "A System and Control theoretic Perspective on Artificial Intelligence Planning Systems", Applied Artificial Intelligence 3:1-32, 1989.
47. J. Rosenblatt, D. Payton, "A Fine-Grained Alternative to the Subsumption Architecture", proceedings of the AAAI Symposium on Robot Navigation, March 1989.
48. D. H. Shin, S. Singh and Wenfan Shi. "A Partitioned Control Scheme for Mobile Robot Path Planning", Proceedings IEEE Conference on Systems Engineering, Dayton, Ohio, August 1991.
49. A. M. Shkel and V. J. Lumelsky, "The joggers problem: accounting for body dynamics in real-time motion planning", in Proceedings of the 1995 International Conference on Intelligent Robots and Systems, Pittsburgh, PA, 5-9 Aug, 1995.
50. S. Singh, et. al, "FastNav: A System for Fast Navigation", Robotics Institute Technical Report CMU-RI-TR-91-20, Carnegie Mellon University, 1991.
51. A. Stentz, "Optimal and Efficient Path Planning for Unknown and Dynamic Environments", CMU Tech report CMU-RI-TR-93-20.
52. A. Stentz and M. Hebert, "A Complete Navigation System for Goal Acquisition in Unknown Environments", Autonomous Robots, Vol. 2, No. 2, August 1995.
53. A. Stentz, "Optimal and Efficient Path Planning for Unknown and Dynamic Environments", International Journal of Robotics and Automation, Vol. 10, No. 3, 1995.
54. J. Spofford and B. Gothard, "Stopping distance analysis of ladar and stereo for unmanned ground vehicles.", Proceed-

ings of the International Society of Optical Engineering, Mobile Robots IX, Boston Mass, 2-4 Nov, 1994.

55. A. Thompson, “The Navigation System of the JPL Robot”, Proceedings of the International Joint Conference for Artificial Intelligence”, 1977, pp749-757

Alonzo Kelly is a project scientist at the Robotics Institute of Carnegie Mellon University. He received his B. A. Sc. in Aerospace engineering from University of Toronto in 1984, his B. Sc. in computer science from York University in 1990, and his Masters and Ph.D. degrees in robotics from Carnegie Mellon University in 1994 and 1996 respectively. His research interests include perception, planning, control, simulation, and operator interfaces for indoor and outdoor mobile robots.



Anthony Stentz is a senior research scientist at the Robotics Institute of Carnegie Mellon University. He received his Ph.D. in computer science from Carnegie-Mellon University in 1989. He received his M.S. in computer science from CMU in 1984 and his B.S. in physics from Xavier University of Ohio in 1982. His research interests include mobile robots, path planning, computer vision, system architecture, and artificial intelligence in the context of fieldworthy robotic systems.

10 Appendix A - List of Symbols

10.1 Lowercase Alphabetics

b ..... baseline

c ..... undercarriage clearance

d ..... disparity

d^* ..... correct disparity

d_{min} ..... minimum disparity

d_{max} ..... maximum disparity

f ..... focal length

h ..... sensor height

p ..... sensor / vehicle nose offset

r ..... wheel radius

s ..... arc length, distance travelled

t ..... time

x ..... crossrange coordinate

y ..... downrange coordinate

z ..... vertical coordinate

z_f ..... front elevation

z_r ..... rear elevation

10.2 Alphabetics

C_d ..... correlation

K_p ..... proportional gain

L ..... vehicle wheelbase

R ..... range

R_{max} ..... maximum range

R_{min} ..... minimum range

S ..... scatter matrix

T ..... time, time interval

V ..... vehicle speed

W ..... vehicle width, swath width

Y ..... groundplane projected range

Y_{min} ..... min groundplane projected range

Y_{max} ..... max groundplane projected range

$X_L\ Y_L$ ..... left camera frame axes

$X_R\ Y_R$ ..... right camera frame axes

10.3 Bold Alphabetics

$\underline{\mathbf{f(x,u)}}$ ..... nonlinear system dynamics model

$\underline{\mathbf{g(x)}}$ ..... terrain following relationship

$\underline{\mathbf{h(x)}}$ ..... hazard model

$\underline{\mathbf{x}}, \underline{\mathbf{x(t)}}, \underline{\mathbf{x_i(t)}}$ ..... state vector, candidate state vector

$\underline{\mathbf{y}}, \underline{\mathbf{y(t)}}, \underline{\mathbf{y_i(t)}}$ ..... output vector, candidate output vector

$\underline{\mathbf{y_{safe}}}$ ..... safety threshold vector

$\underline{\mathbf{u}}, \underline{\mathbf{u(t)}}, \underline{\mathbf{u_i(t)}}$ ..... command vector, candidate cmd vector

$\underline{\mathbf{u}}_{\mathbf{d}}$ .....terrain disturbance vector

$\mathbf{A}$.....system dynamics matrix

$\mathbf{B}$.....input distribution matrix

$\mathbf{X_f}$.....set of mechanically feasible trajectories

$\mathbf{X_a}$.....iset of admissable trajectories

$\mathbf{L[\underline{x}]}$.....goal functional

10.4 Greek Alphabetics

α .....steer angle

$\dot{\beta}$ .....angular velocity (z component)

δ .....normalized disparity

κ .....curvature

ψ .....yaw, pixel azimuth, vehicle yaw

$\dot{\psi}$ .....vehicle yaw rate

θ .....pitch, elevation

ϕ .....roll

10.5 Increments and Differentials

dx, Δx .....crossrange incremental distance

dy, Δy .....downrange incremental distance

dz, Δz .....vertical incremental distance

$d\theta$, $\Delta\theta$ .....pitch/elevation increment or error

$d\psi$, $\Delta\psi$ .....yaw/azimuth increment or error

$\Delta\delta$ .....change in normalized disparity

11 Appendix B - Glossary

Ackerman steering - a steering mechanism, typical of automobiles, where the two front wheels turn together.
active vision - an approach to vision which emphasizes the direction of attention to the relevant parts of the scene.
actuation space - an abstract space consisting of all independent control inputs to a system. For an automobile, this space is spanned by steering, brake, and throttle.
adaptive lookahead - any mechanism which adapts sensory lookahead to the speed and response of the vehicle.
actuator contention problem - the contention of software modules for the control of actuators.
adaptive scan - an algorithm which attempts to make perceptual groundplane resolution homogeneous and isotropic.
adaptive sweep - an algorithm which projects a focus of attention on the groundplane into image space.
adaptive perception - an algorithm based on a focus of attention and a resolution transform which adapts to vehicle speed and response to provide minimum throughput and uniform minimum resolution of perceptual processing.
adaptive regard - an algorithm based on a focus of attention and a resolution transform which adapts to vehicle speed and response to provide minimum throughput and uniform minimum resolution of planning processing.
admissible - the property of being safe.
admissible set - the set of all admissible trajectories.
azimuth - the rotation of something about a vertical axis.
cartesian elevation map - see terrain map.
clothoid - a linear polynomial for curvature expressed in terms of arc length. The trajectory corresponding to this polynomial.
command vector - the vector which contains all command inputs to the vehicle or its controller.
command space - the space of all possible command vectors.
command following problem - the problem of causing a servo-controlled device to follow its commands acceptably well.
command space search - the technique of searching through spatial trajectories that are expressed implicitly in command space.
configuration space - any abstract space of variables which completely determines the positions of all points on a vehicle or mechanism.
crossrange - the horizontal direction transverse to the sensor optical axis.
curvature - the derivative of heading (or vehicle yaw) with respect to distance travelled.
disparity - the difference in the position of two corresponding points in a pair of images.
disparity window - a range of disparities defined by a maximum and a minimum value.
dead reckoning - the process of integrating certain equations which express position and heading in terms of curvature and either distance or time.
dead zone - that region of the local environment which a vehicle is committed to travelling at any particular time.
detection zone - the region of the local environment for which time is almost up to react to hazards.
downrange - the horizontal direction aligned with the sensor optical axis.
dynamically feasible - the property of a spatial trajectory of satisfying the dynamic model of the vehicle.
dynamics constraint - the constraints imposed on a spatial trajectory by the dynamic model of the vehicle.
elevation - the rotation of something about a horizontal axis, or the height of something
feasible - the property of being mechanically feasible.
feasible set - the set of all feasible trajectories.
feedforward - the process of predicting system response in the future in order to modify the command inputs of the moment.
flat terrain assumption - the assumption that the vehicle operates in extremely benign terrain.
focus of attention - in perception and planning, a region of space to which computations which be confined.
forward model - a model of a dynamic system which is of the form state = function of state and command. see inverse model.
free zones - regions of the local environment that the vehicle cannot reach within some time horizon.
goal trajectory - the path to be followed. Generated by the strategic control layer.
goal functional - a measure of the degree of agreement between the goal and a candidate response trajectory.
ground plane - the surface of the terrain when it is assumed to be flat.
goal point - the point on the path which is one lookahead distance away in pure pursuit.
guaranteed detection - the policy of ensuring adequate resolution in computations, sensing, and actuation.
guaranteed localization - the policy of ensuring adequate accuracy in computations, sensing, and actuation.
guaranteed response - the policy of ensuring adequate response time.
guaranteed safety - the policy of guaranteeing vehicle survival.
guaranteed throughput - the policy of ensuring adequate system throughput.
hazard space - an abstract space for which every point represents the safety of the vehicle in terms of multiple hazardous conditions.
hazard vector - a point in hazard space.
horizon zone - that region of the local environment which the vehicle can afford to wait to assess.
image plane - a virtual or real plane in space upon which an imaging sensor forms an image.
impulse turn - a turn from zero curvature to the maximum.
kinematically feasible - the property of a spatial trajectory of satisfying the kinematic limitations of the vehicle model.
kinematics constraint - the constraints imposed on a spatial trajectory by the kinematic model of the vehicle.
inverse model - a model of a dynamic system which is of the form command = function of state. see forward model.
latency - any type of delay in transforming inputs into outputs in a system.
local planning problem - the problem of deciding where to drive

based only on what can be seen at the moment.
local minimum problem - in planning, the problem of avoiding local minima which effectively trap local planners.
multivariate state space system - a state space model whose state vector contains at least two elements.
terrain mapping - the process of generating a map of the environment surrounding the vehicle.
terrain map - a data structure which represents the properties, usually the height, of terrain.

maneuver - the maneuver dynamics aspect of response.
monotone range assumption - the perception assumption that range is monotone in elevation in the image plane.
mechanically feasible - the property of a spatial trajectory of satisfying the kinematic and dynamic limitations of the vehicle model.
motion distortion problem - the distortion of the world model due to unmodeled delays or unmodeled motion.
nonholonomic - the property of a differential constraint that it cannot be integrated.
normalized disparity - disparity divided by baseline. Measured in radians.
output vector - the vector of outputs in a dynamic system.
path - a geometric description of a curve in space (i.e. with time parameter eliminated). See trajectory.
path planning - the process of deciding where to drive. Involves representation, search, and selection of a path.
perception ratio - the ratio of sensor height to measured range. Equal to the tangent of the range pixel incidence angle for flat terrain.
planning window - the focus of perceptual attention expressed in terms of the point of actuation.
predictive control - a method of controlling something better by using feedforward.
pure pursuit - an algorithm for tracking paths characterized by steering towards a point on the path at a distance in front of the vehicle.
range - 3D cartesian distance between the image plane and the scene.
range projection - the projection of range into the groundplane
range gating - a technique of range imaging where all pixels of a particular range window or gate are acquired simultaneously.
range gate - see range window.
range image - an image whose intensity values correspond to the range to the first reflecting surface in the environment.
range window - a region of interest specified in terms of a maximum and a minimum range.
reaction - the computer and sensory processing aspects of response.
reaction time - the time it takes to decide on a course of action and issue the associated commands to the hardware.
region of interest - see focus of attention.
reference point - a distinguished point on the vehicle. Used to track its state.
registration problem - the problem that redundant measurements of the same geometry do not agree.
resolution - the smallest difference that a system can resolve.
response - the total response of the vehicle including the computer and sensory processing and the maneuver dynamics.
response time - the time it takes a vehicle to respond to an external event. Equal to the sum of reaction time and maneuver time.
response-resolution tradeoff - the fact that easing response requirements increases resolution requirements and vice-versa.
reverse turn - a turn from one curvature extreme to the other.
sampling problem - the problem of variation in the shape and size of range pixels when projected onto the ground plane.
small incidence angle assumption - the assumption that the perception ratio is small.
state vector - a point in state space.
state space - an abstract space of all variables of a system necessary to define its response to inputs.
stationary environment assumption - the assumption that the environment is a single rigid body that is stationary, that is, that there are no dynamic obstacles or changes in the terrain etc.
sweep rate - the angular elevation rate of scanning of a range sensor.
tactical control - in the standard model, the layer responsible for dynamics feedforward, obstacle avoidance, and goal-seeking.
trajectory - a parametric description of a curve in space (i.e. with time parameter explicit). See path.
trajectory generation problem - the problem of determining the commands to issue to a vehicle to cause it to follow a given spatial trajectory.
throughput - a measure of amount of information processed per unit time.
throughput problem - the problem of maintaining adequate throughput.
time constant - the coefficient of the first derivative in a first order system.
tunnel vision problem - the problem of inadequate horizontal field of view.
undersampling - the process of sampling below the Nyquist rate.
vertical - the direction aligned with local gravity.
wheelbase - the length of the vehicle measured from back to front wheels.
yaw - rotation about the vertical axis.